Yeow Meng Chee Zhenbo Guo
San Ling Fengjing Shao
Yuansheng Tang Huaxiong Wang
Chaoping Xing (Eds.)

LNCS 6639

Coding and Cryptology

Third International Workshop, IWCC 2011
Qingdao, China, May/June 2011
Proceedings

Springer

Lecture Notes in Computer Science 6639

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison
Lancaster University, UK

Takeo Kanade
Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler
University of Surrey, Guildford, UK

Jon M. Kleinberg
Cornell University, Ithaca, NY, USA

Alfred Kobsa
University of California, Irvine, CA, USA

Friedemann Mattern
ETH Zurich, Switzerland

John C. Mitchell
Stanford University, CA, USA

Moni Naor
Weizmann Institute of Science, Rehovot, Israel

Oscar Nierstrasz
University of Bern, Switzerland

C. Pandu Rangan
Indian Institute of Technology, Madras, India

Bernhard Steffen
TU Dortmund University, Germany

Madhu Sudan
Microsoft Research, Cambridge, MA, USA

Demetri Terzopoulos
University of California, Los Angeles, CA, USA

Doug Tygar
University of California, Berkeley, CA, USA

Gerhard Weikum
Max Planck Institute for Informatics, Saarbruecken, Germany

Yeow Meng Chee Zhenbo Guo San Ling
Fengjing Shao Yuansheng Tang
Huaxiong Wang Chaoping Xing (Eds.)

Coding and Cryptology

Third International Workshop, IWCC 2011
Qingdao, China, May 30 - June 3, 2011
Proceedings

Springer

Volume Editors

Yeow Meng Chee
San Ling
Huaxiong Wang
Chaoping Xing
Nanyang Technological University, School of Physical and Mathematical Sciences
Division of Mathematical Sciences, Singapore 637371
E-mail: ymchee@ntu.edu.sg; lingsan@ntu.edu.sg; hxwang@ntu.edu.sg
xingcp@ntu.edu.sg

Zhenbo Guo
Qingdao University, College of Information Engineering
Shandong, P.R. China 266071
E-mail: gzb@qdu.edu.cn

Fengjing Shao
Qingdao University, Shandong, P.R. China 266071
E-mail: sfj@qdu.edu.cn

Yuansheng Tang
Yangzhou University, School of Mathematical Sciences
Jiangsu, P.R. China 225002
E-mail: ystang@yzu.edu.cn

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-20900-0 e-ISBN 978-3-642-20901-7
DOI 10.1007/978-3-642-20901-7
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011926909

CR Subject Classification (1998): E.3-4, G.2.1, C.2, J.1, D.4.6

LNCS Sublibrary: SL 4 – Security and Cryptology

© Springer-Verlag Berlin Heidelberg 2011
This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.
The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

The Third International Workshop on Coding and Cryptology (IWCC 2011) was held at Qingdao Garden Hotel, Qingdao, China, from May 30 to June 3, 2011. IWCC 2011 was co-organized by Qingdao University, China, and Nanyang Technological University (NTU), Singapore. We acknowledge with gratitude the financial support from Qingdao University.

The biennial International Workshop on Coding and Cryptology (IWCC) aims to bring together many of the world's greatest minds in coding and cryptology to share ideas and exchange knowledge related to advancements in coding and cryptology, amidst an informal setting conducive to interaction and collaboration.

It is well known that fascinating connections exist between coding and cryptology. Therefore this workshop series was organized to facilitate a fruitful interaction and stimulating discourse among experts from these two areas. The first IWCC was held at Wuyi Mountain, Fujian, China, during June 11-15, 2007, and the second IWCC was held during June 1-5, 2009 at Zhangjiajie, Hunan, China.

The proceedings of this workshop consists of 19 technical papers, covering a wide range of topics in coding and cryptology as well as related fields such as combinatorics. All papers were contributed by the invited speakers of the workshop and each paper was carefully reviewed. We are grateful to the external reviewers for their help, which has greatly strengthened the quality of the proceedings.

We would like to express our thanks to Springer for making it possible for the proceedings to be published in the *Lecture Notes in Computer Science* series. We also thank Zhexian Wan for his great encouragement and constant support for this workshop series. We are also grateful to the staff and students from both Qingdao University and Nanyang Technological University for the administrative and technical support they have rendered to the workshop and the proceedings. Special thanks go to Jia Yu for taking care of the website of the workshop, and Hoon Wei Lim for assistance on matters related to LaTeX.

May 2011

Yeow Meng Chee
Zhenbo Guo
San Ling
Fengjing Shao
Yuansheng Tang
Huaxiong Wang
Chaoping Xing

Third International Workshop on Coding and Cryptology (IWCC 2011)

Sponsored and Organized by

Qingdao University, China
Nanyang Technological University, Singapore

General Chair

Fengjing Shao	Qingdao University, China

Chair of Organizing Committee

Zhexian Wan	Academy of Mathamatics and System Sciences, CAS, China

Organizing Committee

Yeow Meng Chee	Nanyang Technological University, Singapore
Xiangguo Cheng	Qingdao University, China
Zhenbo Guo	Qingdao University, China
San Ling	Nanyang Technological University, Singapore
Zhenkuan Pan	Qingdao University, China
Yuansheng Tang	Yangzhou University, China
Huaxiong Wang	Nanyang Technological University, Singapore
Chaoping Xing	Nanyang Technological University, Singapore
Jia Yu	Qingdao University, China

Table of Contents

A Signature Scheme with Efficient Proof of Validity 1
Masayuki Abe and Miyako Ohkubo

Secret-Sharing Schemes: A Survey 11
Amos Beimel

Lattice Codes for the Gaussian Wiretap Channel 47
Jean-Claude Belfiore, Frédérique Oggier, and Patrick Solé

List Decoding for Binary Goppa Codes 62
Daniel J. Bernstein

Faster 2-Regular Information-Set Decoding 81
Daniel J. Bernstein, Tanja Lange, Christiane Peters, and Peter Schwabe

Ideal Secret Sharing Schemes for Useful Multipartite Access Structures 99
Oriol Farràs and Carles Padró

Loiss: A Byte-Oriented Stream Cipher 109
Dengguo Feng, Xiutao Feng, Wentao Zhang, Xiubin Fan, and Chuankun Wu

Secure Message Transmission by Public Discussion: A Brief Survey 126
Juan Garay, Clint Givens, and Rafail Ostrovsky

Variations on Encoding Circuits for Stabilizer Quantum Codes 142
Markus Grassl

Algorithms for the Shortest and Closest Lattice Vector Problems 159
Guillaume Hanrot, Xavier Pujol, and Damien Stehlé

An Experiment of Number Field Sieve over GF(p) of Low Hamming Weight Characteristic 191
Kenichiro Hayasaka and Tsuyoshi Takagi

The Minimum Distance of Graph Codes 201
Tom Høholdt and Jørn Justesen

Local Duality and the Discrete Logarithm Problem 213
Ming-Deh Huang

On the Effects of Pirate Evolution on the Design of Digital Content Distribution Systems 223
Aggelos Kiayias

Arithmetic of Split Kummer Surfaces: Montgomery Endomorphism of Edwards Products 238
David Kohel

A New Family of Quadriphase Sequences with Low Correlation 246
Jie Li, Xiangyong Zeng, and Lei Hu

On the Link of Some Semi-bent Functions with Kloosterman Sums 263
Sihem Mesnager and Gérard Cohen

Locally Decodable Codes: A Brief Survey 273
Sergey Yekhanin

On Relationship of Computational Diffie-Hellman Problem and Computational Square-Root Exponent Problem 283
Fangguo Zhang and Ping Wang

Author Index 295

A Signature Scheme with Efficient Proof of Validity

Masayuki Abe[1] and Miyako Ohkubo[2]

[1] Information Sharing Platform Laboratories, NTT Corporation, Japan
abe.masayuki@lab.ntt.co.jp
[2] NICT, Japan
m.ohkubo@nict.go.jp

Abstract. A signature scheme is presented that, when combined with the Groth-Sahai proof system, can efficiently prove the validity of a committed signature to a message shown in the clear. Compared to the Boneh-Boyen signature scheme, the proposed scheme yields a shorter proof of validity and is based on a more desirable hardness assumption that achieves a better security bound when analyzed in the generic group model.

1 Introduction

1.1 Background

Constructing a secure and efficient digital signature scheme is a central issue in cryptography research. A number of schemes have been introduced in the literature. They are based on different hardness assumptions and different models of computation. In the random oracle model [2], which models hash functions as truly random functions, many efficient schemes have been constructed, e.g., [17,3,15,6,11,14,7]. They are based on simple number-theoretic hardness assumptions, such as the discrete-logarithm (DL for short) assumption, the computational Diffie-Hellman (CDH) assumption, and so on. There are also schemes constructed from similarly simple number theoretic assumptions without random oracles, e.g., [10,8,19,20].

A signature scheme should be compatible with other cryptographic tools so that it can be easily used as a building block for modular constructions of cryptographic protocols. In fact, signature schemes are often combined with non-interactive proof systems in the constructions of privacy-protecting cryptographic protocols, such as verifiably encrypted signatures, group signatures, anonymous credentials, and so on. Currently, the Groth-Sahai (GS) proof system [13] is the only efficient non-interactive proof system whose security is based on standard assumptions. We are therefore seeking efficient signature schemes that are compatible with the GS proofs.

In [1], Abe et al. presented an efficient signature scheme whose public-keys, messages, and signatures are all group elements, and the verification consists of pairing product equations as desired for compatibility with the GS proofs. While

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 1–10, 2011.
© Springer-Verlag Berlin Heidelberg 2011

the scheme provides everything we are seeking, it can be excessive for some purposes. In this paper, we focus on a scenario where the message is exposed in the clear but (some part of) the signature and the public-key are hidden as witnesses of a GS proof. Such a case still happens, for instance, in the construction of group signatures [12,1].

1.2 Boneh-Boyen Signature Scheme

The Boneh-Boyen signature scheme [4] is an efficient signature scheme whose security can be proven without random oracles. It is often used in cryptographic protocol design as it is built over bilinear groups that are compatible with other useful cryptographic tools.

Over a set of bilinear groups that has efficient mapping $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, a Boneh-Boyen signature to message $m \in \mathbb{Z}_p$ is a pair (σ, r) that fulfills the verification equation

$$e(g_1, g_2) = e(\sigma, u\, g_2^m\, v^r), \tag{1}$$

where (g_1, g_2, u, v) is the public-key of the signer. We consider (g_1, g_2) as a common parameter and (u, v) as a public-key for each individual signer. It is strongly existentially unforgeable against adaptive chosen message attacks under the q-strong Diffie-Hellman assumption (SDH) [5]. Given $(g_1, g_1^x, \cdots, g_1^{x^q}, g_2, g_2^x)$, it is infeasible to compute $(c, g_1^{\frac{1}{x+c}})$. In the generic bilinear group model, it is proven that the success probability is bounded by $\mathcal{O}(q\,n^2)/p$ for adversaries that execute n group operations and pairings. To prove one's possession of a valid signature with respect to a committed public-key, relation (1) is shown by the GS proof system with witness (σ, u, v). (Here, we only consider the scenario where r can be exposed in the clear. Note that r is truly random. Thus exposing r does not impact to the secrecy of the individual public-key.)

There are two issues that we would like to address. The first is the efficiency when a signature is combined with the GS proof. Since the pairing in the right side of (1) includes variables in $\mathbb{G}_1$ and $\mathbb{G}_2$ and is not randomizable due to the strong unforgeability, the corresponding GS proof has to prove a costly double-side relation. This is an extra cost since strong unforgeability is unnecessary when the signature is hidden in the GS proof. The second issue is the loose lower bound of SDH. It is far, by a factor of q, from the optimal $\mathcal{O}(n^2)/p$ that applies to DL. While q is supposed to be much smaller than n, it strains the security margin and eventually affects the performance.

1.3 Our Contribution

We present a signature scheme that resolves the above-mentioned two issues. It is regularly, i.e., not strongly, unforgeable and efficient when combined with a GS proof for hiding signatures and public-keys. It is based on a hardness assumption that has almost optimal security bound in the generic bilinear group model and therefore is more desirable than SDH.

The proof for the validity of a signature with our scheme consists of 6 group elements, while that for a Boneh-Boyen signature consists of 14. The drawback is that the common parameter requires a trusted set-up.

The security is proven based on a new q-type assumption that we call the flexible pairing with exponent (FPE) assumption. We show that the FPE has the bound $\mathcal{O}(n^2 + q^2)/p$ in the generic bilinear group model.

2 Preliminaries

2.1 Syntax and Security Notions

We consider signature schemes that consist of four algorithms, (Setup, KeyGen, Sign, Vrify). Setup is a set-up algorithm that generates a public common parameter Λ for given security parameter 1^λ. KeyGen is a key generation algorithm that takes Λ and outputs a pair of a verification-key and a signing-key, (vk, sk). Sign is a signing algorithm that takes as input signing-key sk and message m in a proper domain associated with vk. It outputs a signature σ. Vrify is a verification algorithm that takes vk, σ, and m as input and outputs 1 or 0 to indicate acceptance or rejection, respectively. If the keys and the signature are generated by KeyGen and Sign, Vrify outputs 1.

A signature scheme is existentially unforgeable against adaptive chosen message attacks (EUF-CMA) if, for any polynomial-time adversary $\mathcal{A}$, the probability

$$\Pr\left[\begin{array}{l|l} \Lambda \leftarrow \mathsf{Setup}(1^\lambda) & \\ (vk, sk) \leftarrow \mathsf{KeyGen}(\Lambda) & m^\star \notin Q_m \ \wedge \ 1 \leftarrow \mathsf{Vrify}(vk, m^\star, \sigma^\star) \\ (m^\star, \sigma^\star) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{sig}}}(vk) & \end{array}\right] \tag{2}$$

is negligible in the security parameter λ. $\mathcal{O}_{\mathsf{sig}}$ is the signing oracle that takes m as input and returns $\sigma \leftarrow \mathsf{Sign}(sk, m)$. Q_m is the messages submitted to $\mathcal{O}_{\mathsf{sig}}$. By requiring $(m^\star, \sigma^\star) \notin Q_{m,\sigma}$ where $Q_{m,\sigma}$ is the history of messages and signatures observed by $\mathcal{O}_{\mathsf{sig}}$, we have the notion of strong unforgeability against adaptive chosen message attacks (sEUF-CMA).

2.2 Proof of Validity with Committed Key

We consider a scenario that a prover is to prove his possession of a valid signature on a message with respect to a committed verification-key. Namely, the prover takes (vk, σ) as a witness and computes a Groth-Sahai non-interactive proof π for relation $1 = \mathsf{Vrify}(vk, m, \sigma)$ whose message m is shown in the clear. When vk can be separated into two parts, say $vk = (\Lambda, \upsilon)$, where Λ is common for all signers and υ is unique for individual signers, the common parameter Λ can be placed in the clear. Another case is that σ is separated as $\sigma = (\gamma, \varsigma)$ where γ is independent of vk. Then γ is placed in the clear and only ς plays the role of a witness. The efficiency is evaluated by counting the sum of the sizes of π,

$\mathsf{Commit}(\upsilon)$, γ, and $\mathsf{Commit}(\varsigma)$. Here, $\mathsf{Commit}(X)$ is the commitment of X used in the GS proof. Note that in a real application, the authenticity of the committed verification-key must be guaranteed outside of the proof of validity we consider here.

3 The Scheme

Let $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ be a description of groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ of prime order p equipped with efficient bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ as in [5].

Set-up: Given security parameter λ, select $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$. Also select random generators $g_1 \in \mathbb{G}_1$, g_2, u, and v of $\mathbb{G}_2$, and set $\Lambda = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2, u, v)$ as a common parameter.

Key Generation: Given Λ as input, select $x \in \mathbb{Z}_p$ and compute $a = g_1^x$. Set $vk = (\Lambda, a)$ and $sk = x$.

Signature Generation: Given $m \in \mathbb{Z}_p$, select $r \in \mathbb{Z}_p^*$ and $\alpha \in \mathbb{Z}_p^*$. Compute $s = g_1^\alpha$ and $t = (g_2\ (u^m v^r)^{-x})^{1/\alpha}$. Output $\sigma = (r, s, t)$.

Verification: Given $\sigma = (r, s, t)$ and $m \in \mathbb{Z}_p$, output 1 if $r \neq 0$ and

$$e(g_1, g_2) = e(a, u^m v^r)\ e(s, t) \tag{3}$$

holds. Output 0, otherwise.

The scheme is correct since

$$\begin{aligned} e(a, u^m v^r)\ e(s, t) &= e(g_1^x, u^m v^r)\ e(g_1^\alpha, (g_2\ (u^m v^r)^{-x})^{1/\alpha}) \\ &= e(g_1, (u^m v^r)^x)\ e(g_1, g_2\ (u^m v^r)^{-x})) \\ &= e(g, g_2). \end{aligned}$$

4 Security

Assumption 1 (Flexible Pairing with Exponent: FPE). Let $\mathcal{I}(\lambda)$ be a set of $(\Lambda, g_1, h_1, g_2, h_2)$, where g_1, h_1 are generators of $\mathbb{G}_1$, and g_2, h_2 are generators of $\mathbb{G}_2$. For $I \in \mathcal{I}(\lambda)$, let $\mathcal{R}$ be

$$\mathcal{R} = \{(z, s, t) \in \mathbb{Z}_p^* \times \mathbb{G}_1 \times \mathbb{G}_2 \mid e(g_1, g_2) = e(h_1, h_2^z)\ e(s, t)\}.$$

Given $I \in \mathcal{I}(\lambda)$ and uniformly chosen $R_i \in \mathcal{R}$ for $i = 1, \ldots, q$, it is hard to find another $(z^\star, s^\star, t^\star) \in \mathcal{R}$ with new $z^\star$ not included in any R_i.

Theorem 2. *The proposed scheme is existentially unforgeable against adaptive chosen message attacks if the FPE assumption holds.*

Proof. (sketch) Let A be an adversary that launches an adaptive chosen message attack and successfully outputs a forgery $(m^\dagger, r^\dagger, s^\dagger, t^\dagger)$ for message $m^\dagger$ that was not sent to the signing oracle. There are three cases.

Case 1: $u^{m^\dagger} v^{r^\dagger} = u^m v^r$ holds for (m, r) observed by the signing oracle, or
Case 2: $u^{m^\dagger} v^{r^\dagger} = 1$ happens, or
Case 3: Otherwise.

In Cases 1 and 2, we can use A to solve the discrete logarithm problem between u and v since $\log_u v = (m - m^\dagger)/(r^\dagger - r) \bmod p$ for Case 1 and $\log_u v = -m^\dagger / r^\dagger$ for Case 2. (Note that $r^\dagger \neq 0$ must hold for a successful forgery.) The signing oracle can be simulated perfectly by following the prescribed normal procedure as the signing algorithm does not require $\log_u v$.

In Case 3, we can use A to break Assumption 1. Given an instance I and R_i for $i = 1, \ldots, q$, set $a = h_1$, $u = h_2^\alpha$, and $v = h_2$ with random $\alpha \in \mathbb{Z}_p^*$. The signing oracle can be simulated by picking a fresh R_i and computing r that fulfills relation $z_i = \alpha m + r$ for given m. (The simulation fails if $r = 0$, which happens with negligible probability.) This yields a valid signature (r, s_i, t_i) for m. From the output of A, a new answer $(z^\star, s^\star, t^\star) = (\alpha m^\dagger + r^\dagger, s^\dagger, t^\dagger)$ can be extracted.

Since the discrete-logarithm assumption is implied by FPE, the signature scheme is secure under FPE as stated. ■

Theorem 3. *Assumption 1 holds in the generic bilinear group model. For any adversary that performs at most n group operations and pairings, the probability of breaking the assumption is upper bounded by $\mathcal{O}(n^2 + q^2)/p$.*

Proof. For the sake of simplicity, we prove the theorem in the symmetric bilinear group setting where $\mathbb{G}_1$ and $\mathbb{G}_2$ are the same. (In the generic model, this implies that the same holds for the asymmetric case that is more restrictive to the adversaries.) In this setting g_1 and g_2 are independent random bases. By $\mathbb{G}$ we denote the base group, $\mathbb{G} = \mathbb{G}_1 = \mathbb{G}_2$. Let $\tilde{g}$ denote a generator of $\mathbb{G}$. For $X \in \mathbb{G}$, let $\hat{X}$ denote index of X, i.e., $\hat{X} = \log_{\tilde{g}} X \in \mathbb{Z}_p$. Define $\hat{X}$ for $X \in \mathbb{G}_T$ in the same manner with respect to base $e(\tilde{g}, \tilde{g})$, respectively.

At the beginning, adversary A is given group elements g_1, g_2, h_1, and h_2. For $i = 1, \ldots, q$, it is also given $z_i \in \mathbb{Z}_p^*$, s_i and t_i, where

$$\hat{s}_i = \alpha_i (\hat{g_1}\, \hat{g_2} - z_i\, \hat{h_1}\, \hat{h_2}) \text{ and} \tag{4}$$

$$\hat{t}_i = 1/\alpha_i \tag{5}$$

for some $\alpha_i \in \mathbb{Z}_p^*$. (Note that we exclude the case where $\hat{s}_i = 0$ or $\hat{t}_i = 0$ where trivial solutions exist. It happens only with negligible probability, $2q/p$.) We call $(\hat{g_1}, \hat{g_2}, \hat{h_1}, \hat{h_2}, \hat{s}_1, \hat{t}_1, \ldots, \hat{s}_q, \hat{t}_q)$ initial indices. Observe that

$$\hat{s}^\star \hat{t}^\star = \hat{g_1}\, \hat{g_2} - z^\star\, \hat{h_1}\, \hat{h_2} \tag{6}$$

holds. Since $\hat{s}^\star$ and $\hat{t}^\star$ must be representable as linear combinations of the initial indices, we can write

$$\hat{s}^\star = c_1\hat{g_1} + c_2\hat{g_2} + c_3\hat{h_1} + c_4\hat{h_2} + \sum_{i=1}^{q}\left(c_{5+2(i-1)}\alpha_i(\hat{g_1}\,\hat{g_2} - z_i\,\hat{h_1}\,\hat{h_2}) + c_{6+2(i-1)}/\alpha_i\right) \text{ and} \tag{7}$$

$$\hat{t}^\star = d_1\hat{g_1} + d_2\hat{g_2} + d_3\hat{h_1} + d_4\hat{h_2} + \sum_{i=1}^{q}\left(d_{5+2(i-1)}\alpha_i(\hat{g_1}\,\hat{g_2} - z_i\,\hat{h_1}\,\hat{h_2}) + d_{6+2(i-1)}/\alpha_i\right) \tag{8}$$

for some constants c_i and d_i for $i = 1, \ldots, 4 + 2q$.

Suppose that $c_1 \neq 0$. Then $d_2 \neq 0$ and $d_i = 0$ for all $i \neq 2$ to avoid terms of g_1 not included in (6). This also leads to $c_i = 0$ for all $i \neq 1$. However, it results in $z^\star = 0$, which is not allowed. Thus $c_1 = 0$. Similarly, we can show that c_2, c_3, c_4, d_1, d_2, d_3, and d_4 are zero. Thus we have

$$\hat{s}^\star = \sum_{i=1}^{q}\left(c_{5+2(i-1)}\alpha_i(\hat{g_1}\,\hat{g_2} - z_i\,\hat{h_1}\,\hat{h_2}) + c_{6+2(i-1)}/\alpha_i\right), \text{ and} \tag{9}$$

$$\hat{t}^\star = \sum_{i=1}^{q}\left(d_{5+2(i-1)}\alpha_i(\hat{g_1}\,\hat{g_2} - z_i\,\hat{h_1}\,\hat{h_2}) + d_{6+2(i-1)}/\alpha_i\right). \tag{10}$$

Next suppose that i exists such that $c_{5+2(i-1)} \neq 0$ and $c_{6+2(i-1)} \neq 0$. To avoid terms in α_i^2 and $1/\alpha_i^2$, $d_{5+2(i-1)} = d_{6+2(i-1)} = 0$ must hold. However, it results in including terms in α_i and $1/\alpha_i$ or forcing $\hat{t}^\star = 0$, which are not allowed. Thus only one of $c_{5+2(i-1)}$ or $c_{6+2(i-1)}$ can be non-zero. Without loss of generality, we assume $c_{5+2(i-1)} \neq 0$ and $c_{6+2(i-1)} = 0$. This implies that $d_{5+2(i-1)} = 0$ and $d_{6+2(i-1)} \neq 0$. Thus we have

$$\hat{s}^\star = \sum_{i\in I} c_{5+2(i-1)}\alpha_i(\hat{g_1}\,\hat{g_2} - z_i\,\hat{h_1}\,\hat{h_2}) + \sum_{i\in J} c_{6+2(i-1)}/\alpha_i, \text{ and} \tag{11}$$

$$\hat{t}^\star = \sum_{i\in J} d_{5+2(i-1)}\alpha_i(\hat{g_1}\,\hat{g_2} - z_i\,\hat{h_1}\,\hat{h_2}) + \sum_{i\in I} d_{6+2(i-1)}/\alpha_i \tag{12}$$

for distinct sets I and J where $I \cup J \subseteq \{1, \ldots, q\}$.

Suppose that both I and J are non-empty. Then, $\hat{s}^\star\hat{t}^\star$ includes terms in $\alpha_i\alpha_j$ for $i \in I$ and $j \in J$. Thus either I or J must be empty. Without loss of generality, we assume $J = \emptyset$. We now have

$$\hat{s}^\star = \sum_{i\in I} c_{5+2(i-1)}\alpha_i(\hat{g_1}\,\hat{g_2} - z_i\,\hat{h_1}\,\hat{h_2}), \text{ and} \tag{13}$$

$$\hat{t}^\star = \sum_{i\in I} d_{6+2(i-1)}/\alpha_i. \tag{14}$$

Suppose that I includes at least two distinct indices, say i and j. Then $\hat{s}^\star\hat{t}^\star$ includes terms in α_i/α_j. Thus I can include only one index. We thus have

$$\hat{s}^\star = c_{5+2(i-1)}\alpha_i(\hat{g_1}\,\hat{g_2} - z_i\,\hat{h_1}\,\hat{h_2}), \text{ and} \tag{15}$$

$$\hat{t}^\star = d_{6+2(i-1)}/\alpha_i \tag{16}$$

for some i. Since $\hat{s}^\star\hat{t}^\star = c_{5+2(i-1)}d_{6+2(i-1)}(\hat{g_1}\,\hat{g_2} - z_i\,\hat{h_1}\,\hat{h_2}) = \hat{g_1}\,\hat{g_2} - z^\star\,\hat{h_1}\,\hat{h_2}$, we have $c_{5+2(i-1)}d_{6+2(i-1)} = 1$. However, it results in $z^\star = z_i$. Accordingly, (6) holds identically only when $z^\star \in \{z_1, \ldots, z_q\}$.

Finally, we calculate the success probability of simulating elements in the groups. The simulation of $\mathbb{G}$ is successful if two distinct group elements, say X and X', coincidentally evaluate to the same value with a random assignment to the variables. Recall that every element X in $\mathbb{G}$ is represented by an index in the form of (7). Therefore,

$$\begin{aligned}\hat{X} - \hat{X}' =& c_1\hat{g_1} + c_2\hat{g_2} + c_3\hat{h_1} + c_4\hat{h_2} \\ &+ \sum_{i=1}^{q}\left(c_{5+2(i-1)}\alpha_i(\hat{g_1}\,\hat{g_2} - z_i\,\hat{h_1}\,\hat{h_2}) + c_{6+2(i-1)}/\alpha_i\right)\end{aligned} \tag{17}$$

for some constants c_i. We consider the probability that $\hat{X} - \hat{X}' = 0$ happens with a random assignment to the variables in the right side of (17).

Before proceeding, we provide a lemma that extends Schwartz's in [18] to a special form of Laurent polynomials. [1]

Lemma 4. *Let F be a Laurent polynomial of form*

$$F = \sum_i \left(\psi_i\alpha_i^{\ell_i} + \mu_i + \xi_i\alpha_i^{-d_i}\right). \tag{18}$$

Here, every α_i is a variable over $\mathbb{Z}_p^*$, and ψ_i and ξ_i are multi-variate polynomials over $\mathbb{Z}_p$, where $\deg_{\alpha_i}(\psi_i) = \deg_{\alpha_i}(\xi_i) = 0$, $\psi_i \neq 0$, and $-d_i < \deg_{\alpha_i}(\mu_i) < \ell_i$. Then the probability that $F = 0$ happens with a random assignment to the variables is at most

$$\min_i(\deg(\psi_i) + \ell_i + d_i)/(p-1). \tag{19}$$

Proof. Let $i^\star$ be the index where $\deg(\psi_{i^\star}) + \ell_{i^\star} + d_{i^\star} = \min_i(\deg(\psi_i) + \ell_i + d_i)$ holds. Let $F_{i^\star} = {\alpha_{i^\star}}^{d_{i^\star}} \cdot F$. Since $\alpha_{i^\star}$ is not zero, $\Pr[F = 0] = \Pr[F_{i^\star} = 0]$. Note that $F_{i^\star}$ is a polynomial of degree $\ell_{i^\star} + d_{i^\star}$ in $\alpha_{i^\star}$. For every assignment to the variables other than $\alpha_{i^\star}$, if $\psi_{i^\star} \neq 0$, there are at most $\ell_{i^\star} + d_{i^\star}$ values of $\alpha_{i^\star}$ with which $F_{i^\star}$ evaluates to zero. Then, $\psi_{i^\star} = 0$ happens with probability of at most $\deg(\psi_{i^\star})/p$ from Schwartz's Lemma. By applying the union bound, we have $\Pr[F_{i^\star} = 0] \leq (\ell_{i^\star} + d_{i^\star})/(p-1) + \deg(\psi_{i^\star})/p < (\deg(\psi_{i^\star}) + \ell_{i^\star} + d_{i^\star})/(p-1)$ as stated. □

[1] [16] shows a bound for general Laurent polynomials. It however is much looser when applied to our specific case.

Formula (17) is in the form of (18) with $\ell_i = d_i = 1$ for all i. Thus $\ell_{i^\star} = d_{i^\star} = 1$ and $\deg(\psi_{i^\star}) = 2$, and the probability that $\hat{X} - \hat{X}' = 0$ is at most $\frac{4}{p-1}$ due to Lemma 4. Since at most $4 + 2q + n$ elements of $\mathbb{G}$ can appear, the probability that $\hat{X} - \hat{X}' = 0$ happens for any combination of X and X' is upper bounded by $\binom{4+2q+n}{2}\frac{4}{p-1} \in \frac{\mathcal{O}(n^2+q^2)}{p}$.

We next consider the case of $\mathbb{G}_T$. Since every element X in $\mathbb{G}_T$ is in a linear combination of a product of two indices in the form of (17), $\hat{X} - \hat{X}'$ is in the form of (18) with $\ell_i = d_i = 2$ for all i. Hence we have $\ell_{i^\star} = d_{i^\star} = 2$ and $\deg(\psi_{i^\star}) = 4$. From Lemma 4, the probability that $\hat{X} - \hat{X}' = 0$ is at most $\frac{8}{p-1}$. Since the number of elements in $\mathbb{G}$ is at most n, the probability that $\hat{X} - \hat{X}' = 0$ happens for any combination of X and X' is at most $\frac{8n^2}{p-1}$ which is also in $\frac{\mathcal{O}(n^2+q^2)}{p}$. ■

5 Comparison

As observed in the previous section, the underlying assumption for our scheme provides better security assessment, by a factor of $\log q$, than that for the Boneh-Boyen signature scheme. In practice, it saves $\log q$ bits from the size of the groups and results in gaining slightly better performance in group operations. In this section, however, we compare the efficiency for the case when the schemes are built on the same bilinear groups.

Consider the case where a signature and a verification-key are committed and the validity of the signature is to be proven with the GS proof system. Since r and t in a signature distribute uniformly over $\mathbb{Z}_p^*$ and $\mathbb{G}_2^*$, they are independent of verification-key a. Thus it is safe to expose r and t in the clear. Accordingly, the relation to prove is

$$e(g_1, g_2) = e(\underline{a}, u^m v^r)\, e(\underline{s}, t)$$

where a and s are the witnesses. This pairing product equation is one-sided, i. e., the variables exist only in $\mathbb{G}_1$, and has advantage in combination with the GS proof system. In the case of the Boneh-Boyen signature scheme, the relation is

$$e(g_1, g_2) = e(\underline{\sigma}, \underline{u}\, \underline{g_2^m}\, \underline{v}^r)$$

which is a double-sided pairing product equation that has variables in both $\mathbb{G}_1$ and $\mathbb{G}_2$.

Table 1 shows the number of group elements in the proof statement, committed witnesses, and a proof when the Groth-Sahai proof system is used as a witness indistinguishable (WI) proof of knowledge over a type-III (asymmetric) and type-I (symmetric) bilinear group setting. (For the properties of these bilinear groups see [9].) For simplicity, we assume the sizes of elements in $\mathbb{G}_1$ and $\mathbb{G}_2$ are the same, and every element is counted as 1.

With both schemes, the zero-knowledge property can be obtained by additionally committing to g_1 and proving that the commitment opens to g_1 by using multi-scalar multiplication equation. This extra part costs, in Type-III setting, 2 group elements in $\mathbb{G}_1$ for committing to g_1 and 2 elements of $\mathbb{Z}_p$ for its proof, and 3 group elements in $\mathbb{G}$ and 3 elements of $\mathbb{Z}_p$ in Type-I setting.

Table 1. Storage cost for proving validity of committed signature and verification-key with GS witness indistinguishable proofs

	Common Parameter	Clear Part	Witness	Number of Elements for Commit & Proof	
				Type-III ($\mathbb{G}_1$, $\mathbb{G}_2$)	Type-I
BB04[4]	(g_1, g_2)	(m, r)	(σ, u, v)	6, 8	18
Ours	(g_1, g_2, u, v)	(m, r, t)	(a, s)	6, 0	9

6 Conclusion

A signature scheme that can be efficiently combined with the GS proof system was presented. When a message is sent in the clear but the verification key and the signature are to be hidden in a commitment, the GS proof for the validity of the hidden signature can be shorter than that of the Boneh-Boyen signatures. We proved the security based on a novel non-interactive assumption and argue its plausibility in the generic bilinear group model. The analysis presents a bound that is almost the same as that for the discrete logarithm problem.

Such a signature scheme is useful for modular construction of many privacy-protecting cryptographic protocols such as group signatures. A potential shortcoming would be that it requires a trusted set-up for the common parameters.

References

1. Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K., Ohkubo, M.: Structure-preserving signatures and commitments to group elements. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 209–236. Springer, Heidelberg (2010)
2. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: First ACM Conference on Computer and Communication Security. Association for Computing Machinery, pp. 62–73 (1993)
3. Bellare, M., Rogaway, P.: The exact security of digital signatures - how to sign with RSA and rabin. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 399–416. Springer, Heidelberg (1996)
4. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
5. Boneh, D., Boyen, X.: Short signatures without random oracles and the sdh assumption in bilinear groups. Journal of Cryptology 21(2), 149–177 (2008)
6. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the weil pairing. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 514–532. Springer, Heidelberg (2001)
7. Chevallier-Mames, B.: An efficient CDH-based signature scheme with a tight security reduction. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 511–526. Springer, Heidelberg (2005)
8. Cramer, R., Shoup, V.: Signature schemes based on the strong RSA assumptions. In: ACM CCS 1999, pp. 46–51 (1999)

9. Galbraith, S., Paterson, K., Smart, N.: Pairings for cryptographers. Technical Report 2006/165, IACR ePrint archive (2006)
10. Gennaro, R., Halevi, S., Rabin, T.: Secure hash-and-sign signatures without the random oracle. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 123–139. Springer, Heidelberg (1999)
11. Goh, E.-J., Jarecki, S.: A signature scheme as secure as the diffie-hellman problem. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 401–415. Springer, Heidelberg (2003)
12. Groth, J.: Fully anonymous group signatures without random oracles. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 164–180. Springer, Heidelberg (2007)
13. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008); Full version available: IACR ePrint Archive 2007/155
14. Katz, J., Wang, N.: Efficiency improvements for signature schemes with tight security reductions. In: ACM Conference on Computer and Communications Security, pp. 155–164. ACM, New York (2003)
15. Pointcheval, D., Stern, J.: Security arguments for digital signatures and blind signatures. Journal of Cryptology 13(3), 339–360 (2000)
16. A. Rupp, G. Leander, E. Bangerter, A.-R. Sadeghi, and A. W. Dent. Sufficient conditions for intractability over black-box groups: Generic lower bounds for generalizaed dl and dh problems. IACR ePrint Archive 2007/360, 2007.
17. Schnorr, C.P.: Efficient signature generation for smart cards. Journal of Cryptology 4(3), 239–252 (1991)
18. Schwartz, J.T.: Fast probabilistic algorithms for verification of polynomial identities. Journal of the ACM 27(4) (1980)
19. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
20. Waters, B.: Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009)

Secret-Sharing Schemes: A Survey*

Amos Beimel

Dept. of Computer Science, Ben-Gurion University of the Negev, Beer-Sheva, Israel

Abstract. A secret-sharing scheme is a method by which a dealer distributes shares to parties such that only authorized subsets of parties can reconstruct the secret. Secret-sharing schemes are an important tool in cryptography and they are used as a building box in many secure protocols, e.g., general protocol for multiparty computation, Byzantine agreement, threshold cryptography, access control, attribute-based encryption, and generalized oblivious transfer.

In this survey, we describe the most important constructions of secret-sharing schemes; in particular, we explain the connections between secret-sharing schemes and monotone formulae and monotone span programs. We then discuss the main problem with known secret-sharing schemes – the large share size, which is exponential in the number of parties. We conjecture that this is unavoidable. We present the known lower bounds on the share size. These lower bounds are fairly weak and there is a big gap between the lower and upper bounds. For linear secret-sharing schemes, which is a class of schemes based on linear algebra that contains most known schemes, super-polynomial lower bounds on the share size are known. We describe the proofs of these lower bounds. We also present two results connecting secret-sharing schemes for a Hamiltonian access structure to the NP vs. coNP problem and to a major open problem in cryptography – constructing oblivious-transfer protocols from one-way functions.

1 Introduction

Secret-sharing schemes are a tool used in many cryptographic protocols. A secret-sharing scheme involves a dealer who has a secret, a set of n parties, and a collection $\mathcal{A}$ of subsets of parties called the access structure. A secret-sharing scheme for $\mathcal{A}$ is a method by which the dealer distributes shares to the parties such that: (1) any subset in $\mathcal{A}$ can reconstruct the secret from its shares, and (2) any subset not in $\mathcal{A}$ cannot reveal any partial information on the secret. Originally motivated by the problem of secure information storage, secret-sharing schemes have found numerous other applications in cryptography and distributed computing, e.g., Byzantine agreement [54], secure multiparty computations [13,24,28], threshold cryptography [31], access control [52], attribute-based encryption [40,68], and generalized oblivious transfer [59,65].

* Research supported by ISF grant 938/09 and by the Frankel Center for Computer Science.

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 11–46, 2011.
© Springer-Verlag Berlin Heidelberg 2011

Example 1 (Attribute Based Encryption). Public-key encryption is a powerful mechanism for protecting the confidentiality of stored and transmitted information. Nowadays, in many applications there is a provider that wants to share data according to some policy based on user's credentials. In an attributed-based encryption system, presented by Sahai and Waters [57], each user has a set of attributes (i.e., credentials), and the provider will grant permission to decrypt the message if some predicate of the attributes holds (e.g., a user can decode an e-mail if she is a "FRIEND" and "IMPORTANT"). In [40,68], it is shown that if the predicate can be described by an access structure that can be implemented by an efficient linear secret-sharing scheme, then there is an efficient attribute-based encryption system for this predicate.

Secret-sharing schemes were introduced by Blakley [17] and Shamir [58] for the threshold case, that is, for the case where the subsets that can reconstruct the secret are all the sets whose cardinality is at least a certain threshold. Secret-sharing schemes for general access structures were introduced and constructed by Ito, Saito, and Nishizeki [45]. More efficient schemes were presented in, e.g., [14,61,21,46,16]. Specifically, Benaloh and Leichter [14] proved that if an access structure can be described by a small *monotone formula* then it has an efficient perfect secret-sharing scheme. This was generalized by Karchmer and Wigderson [46], who showed that if an access structure can be described by a small *monotone span program* then it has an efficient scheme (a special case of this construction appeared before in [21]).

A major problem with secret-sharing schemes is that the shares' size in the best known secret-sharing schemes realizing general access structures is exponential in the number of parties in the access structure. Thus, the known constructions for general access structures are impractical. This is true even for explicit access structures (e.g., access structures whose characteristic function can be computed by a small uniform circuit). On the other hand, the best known lower bounds on the shares' size for sharing a secret with respect to an access structure (e.g., in [23,29]) are far from the above upper bounds. The best lower bound was proved by Csirmaz [29], proving that, for every n, there is an access structure with n parties such that sharing ℓ-bit secrets requires shares of length $\Omega(\ell n/\log n)$. The question if there exist more efficient schemes, or if there exist access structures that do not have (space) efficient schemes remains open. The following is a widely believed conjecture (see, e.g., [3]):

Conjecture 1. There exists an $\epsilon > 0$ such that for every integer n there is an access structure with n parties, for which every secret-sharing scheme distributes shares of length exponential in the number of parties, that is, $2^{\epsilon n}$.

Proving (or disproving) this conjecture is one of the most important open questions concerning secret sharing. No major progress on proving or disproving this conjecture has been obtained in the last 16 years. It is not known how to prove that there exists an access structure that requires super-polynomial shares (even for an implicit access structure).

Most previously known secret-sharing schemes are *linear*. In a linear scheme, the secret is viewed as an element of a finite field, and the shares are obtained

by applying a linear mapping to the secret and several independent random field elements. For example, the schemes of [58,17,45,14,61,16,46] are all linear. For many application, the linearity is important, e.g., for secure multiparty computation as will be described in Section 4. Thus, studying linear secret-sharing schemes and their limitations is important. Linear secret-sharing schemes are equivalent to monotone span programs, defined by [46]. Super-polynomial lower bounds for monotone span programs and, therefore, for linear secret-sharing schemes were proved in [5,2,36].

In this survey we will present two unpublished results of Rudich [56]. Rudich considered a Hamiltonian access structure; the parties in this access structure are edges in a complete undirected graph, and a set of edges (parties) is authorized if it contains a Hamiltonian cycle.[1] Rudich proved that if $NP \neq coNP$, then this access structure does not have a secret-sharing scheme in which the sharing of the secret can be done by a polynomial-time algorithm. As efficient sharing of secrets is essential in applications of secret-sharing, Rudich's results implies that there is no practical scheme for the Hamiltonian access structure. Furthermore, Rudich proved that if one-way functions exist and if the Hamiltonian access structure has a computational secret-sharing scheme (with efficient sharing and reconstruction), then efficient protocols for oblivious transfer exists. Thus, constructing a computational secret-sharing scheme for the Hamiltonian access structure based on one-way functions will solve a major open problem in cryptography, i.e., using Impagliazzo's terminology [43], it will prove that Minicrypt = Cryptomania.

1.1 Organization

This survey is intended for readers with some background in cryptography and complexity. When possible, we try to give the required definitions. The rest of the survey is organized as follows. In Section 2 we define secret-sharing schemes, giving two definitions and proving that they are equivalent. In Section 3, we present constructions of secret-sharing schemes. In Section 4, we show how to construct secure multiparty protocols for general functions (in the honest-but-curious model) using secret-sharing schemes. In Section 5, we discuss lower bounds for secret-sharing schemes and present the best known lower bounds for general secret-sharing schemes and linear secret-sharing schemes. In Section 6, we present the unpublished results of Rudich. Finally, in Section 7, we summarize this survey and mention the most important open problems for secret sharing.

2 Definitions

In this section we define secret-sharing schemes. We supply two definitions and argue that they are equivalent.

[1] The results of Rudich apply to other monotone NP-complete problem as well, e.g., the clique problem.

Definition 1 (Access Structure, Distribution Scheme). *Let $\{p_1, \ldots, p_n\}$ be a set of parties. A collection $\mathcal{A} \subseteq 2^{\{p_1,\ldots,p_n\}}$ is* monotone *if $B \in \mathcal{A}$ and $B \subseteq C$ imply that $C \in \mathcal{A}$. An* access structure *is a monotone collection $\mathcal{A} \subseteq 2^{\{p_1,\ldots,p_n\}}$ of non-empty subsets of $\{p_1, \ldots, p_n\}$. Sets in $\mathcal{A}$ are called* authorized*, and sets not in $\mathcal{A}$ are called* unauthorized*.*

A distribution scheme *$\Sigma = \langle \Pi, \mu \rangle$ with domain of secrets K is a pair, where μ is a probability distribution on some finite set R called the set of random strings and Π is a mapping from $K \times R$ to a set of n-tuples $K_1 \times K_2 \times \cdots \times K_n$, where K_j is called the* domain of shares *of p_j. A dealer distributes a secret $k \in K$ according to Σ by first sampling a random string $r \in R$ according to μ, computing a vector of* shares *$\Pi(k, r) = (s_1, \ldots, s_n)$, and privately communicating each share s_j to party p_j. For a set $A \subseteq \{p_1, \ldots, p_n\}$, we denote $\Pi(s, r)_A$ as the restriction of $\Pi(s, r)$ to its A-entries.*

The information ratio *of a distribution scheme is $\frac{\max_{1 \le j \le n} \log |K_j|}{\log |K|}$. The* average information ratio *of a distribution scheme is $\frac{\sum_{1 \le j \le n} \log |K_j|}{n \cdot \log |K|}$.* [2]

We start with a definition of secret-sharing as given in [26,4,12].

Definition 2 (Secret Sharing). *Let K be a finite set of secrets, where $|K| \ge 2$. A distribution scheme $\langle \Pi, \mu \rangle$ with domain of secrets K is a* secret-sharing scheme *realizing an access structure $\mathcal{A}$ if the following two requirements hold:*

Correctness. *The secret k can be reconstructed by any authorized set of parties. That is, for any set $B \in \mathcal{A}$ (where $B = \{p_{i_1}, \ldots, p_{i_{|B|}}\}$), there exists a* reconstruction function $\text{RECON}_B : K_{i_1} \times \ldots \times K_{i_{|B|}} \to K$ *such that for every $k \in K$,*

$$\Pr[\, \text{RECON}_B(\Pi(k, r)_B) = k \,] = 1. \tag{1}$$

Perfect Privacy. *Every unauthorized set cannot learn anything about the secret (in the information theoretic sense) from their shares. Formally, for any set $T \notin \mathcal{A}$, for every two secrets $a, b \in K$, and for every possible vector of shares $\langle s_j \rangle_{p_j \in T}$:*

$$\Pr[\, \Pi(a, r)_T = \langle s_j \rangle_{p_j \in T} \,] = \Pr[\, \Pi(b, r)_T = \langle s_j \rangle_{p_j \in T} \,]. \tag{2}$$

Remark 1. In the above definition, we required correctness with probability 1 and perfect privacy: for every two secrets a, b the distributions $\Pi(a, r)_T$ and $\Pi(a, r)_T$ are identical. We can relax these requirements and require that the correctness holds with high probability and that the statistical distance between $\Pi(a, r)_T$ and $\Pi(a, r)_T$ is small. Schemes that satisfy these relaxed requirements are called statistical secret-sharing schemes. For example, such schemes are designed in [16].

[2] In the secret sharing literature it is also common to use the term *information rate*, which is the inverse of the information ratio.

We next define an alternative definition of secret-sharing schemes originating in [47,23]; this definition uses the entropy function. For this definition we assume that there is some known probability distribution on the domain of secrets K. Any probability distribution on the domain of secrets, together with the distribution scheme Σ, induces, for any $A \subseteq \{p_1, \ldots, p_n\}$, a probability distribution on the vector of shares of the parties in A. We denote the random variable taking values according to this probability distribution on the vector of shares of A by S_A, and by S the random variable denoting the secret. The privacy in the alternative definition requires that if $T \notin \mathcal{A}$, then the random variables S and S_T are independent.

As traditional in the secret sharing literature, we formalize the above two requirements using the entropy function. The support of a random variables X is the set of all values x such that $\Pr[X = x] > 0$. Given a random variable X, the *entropy* of X is defined as $H(X) \stackrel{\text{def}}{=} \sum \Pr[X = x] \log 1/\Pr[X = x]$, where the sum is taken over all values x in the support of X, i.e., all values x such that $\Pr[X = x] > 0$. It holds that $0 \leq H(X) \leq \log |\text{SUPPORT}(X)|$. Intuitively, $H(X)$ measures the amount of uncertainty in X where $H(X) = 0$ if X is deterministic, i.e., there is a value x such that $\Pr[X = x] = 1$, and $H(X) = \log |\text{SUPPORT}(X)|$ if X is uniformly distributed over $\text{SUPPORT}(X)$. Given two random variables X and Y we consider their concatenation XY and define the *conditional entropy* as $H(X|Y) \stackrel{\text{def}}{=} H(XY) - H(Y)$. It holds that $0 \leq H(X|Y) \leq H(X)$; two random variables X and Y are independent iff $H(X|Y) = H(X)$ and the value of Y implies the value of X iff $H(X|Y) = 0$. For more background on the entropy function, the reader may consult [27].

Definition 3 (Secret Sharing – Alternative Definition). *We say that a distribution scheme is a* secret-sharing scheme realizing an access structure $\mathcal{A}$ with respect to a given probability distribution on the secrets, denoted by a random variable S, if the following conditions hold.

Correctness. For every authorized set $B \in \mathcal{A}$,

$$H(S|S_B) = 0. \tag{3}$$

Privacy. For every unauthorized set $T \notin \mathcal{A}$,

$$H(S|S_T) = H(S). \tag{4}$$

Definition 2 and Definition 3 are equivalent, as proved below in Claim 1. The advantage of Definition 2 is that it does not assume that there is a probability distribution on the secrets and that this distribution is known. Furthermore, Definition 2 can be generalized to statistical secret sharing and computational secret sharing. On the other hand, Definition 3 is more convenient for proving lower bounds. Thus, the equivalence of the definitions allows choosing the more suitable definition for the specific task.

Furthermore, the equivalence of the definitions allows proving a result of Blundo et al. [20] that the privacy of a scheme according to Definition 3 is

actually independent of the distribution: If a scheme realizes an access structure with respect to one distribution on the secrets, then it realizes the access structure with respect to any distribution with the same support.

Claim 1. *The following claims are equivalent for a distribution scheme Σ:*

1. *The scheme Σ is secure according to Definition 2.*
2. *There is some distribution on the secrets with support K (that is, $\Pr[S = a] > 0$ for every $a \in K$) such that the scheme is secure according to Definition 3.*
3. *For every distribution on the secrets whose support is contained in K, the scheme is secure according to Definition 3.*

Proof. We first show that (1) implies (3) (and, hence, (2)). Let $\Sigma = \langle \Pi, \mu \rangle$ be a secret-sharing scheme which is private according to Definition 2, and let S be random variable distributed according to some distribution over K. Thus, for any set $T \notin \mathcal{A}$, any secret $a \in K$, and any shares $\langle s_j \rangle_{p_j \in T}$ for the parties in T,

$$\begin{aligned}
\Pr[\, S_T = \langle s_j \rangle_{p_j \in T} | S = a \,] &= \Pr[\, \Pi(a, r)_T = \langle s_j \rangle_{p_j \in T} \,] \\
&= \sum_{b \in K} \Pr[S = b] \cdot \Pr[\, \Pi(b, r)_T = \langle s_j \rangle_{p_j \in T} \,] \qquad (5) \\
&= \sum_{b \in K} \Pr[S = b] \cdot \Pr[\, S_T = \langle s_j \rangle_{p_j \in T} | S = b \,] \\
&= \Pr[\, S_T = \langle s_j \rangle_{p_j \in T} \,], \qquad (6)
\end{aligned}$$

where the equality in (5) follows from (2). In other words, by (6), S_T and S are independent random variables, and, by the properties of the entropy function, $H(S|S_T) = H(S)$, thus, the scheme is private according to Definition 3 with respect to this distribution on S.

Now assume that $\Sigma = \langle \Pi, \mu \rangle$ is a secret-sharing scheme which is private according to Definition 3 for some fixed distribution on the secrets with support K, that is, assume that (2) holds. For any set $T \notin \mathcal{A}$, the random variables S_T and S are independent, and, in particular, for every pair of secrets $a, b \in K$, and every shares $\langle s_j \rangle_{p_j \in T}$

$$\Pr_r[\, \Pi(a, r)_T = \langle s_j \rangle_{p_j \in T} \,] = \Pr_{r,k}[\, \Pi(k, r)_T = \langle s_j \rangle_{p_j \in T} \,] = \Pr_r[\, \Pi(b, r)_T = \langle s_j \rangle_{p_j \in T} \,],$$

where the first and last probabilities are for fixed secrets and are taken over the randomness of Π, and the middle probability is over both the randomness of Π and the secret k chosen according to the fixed distribution. Thus, the scheme is secure according to Definition 2. □

3 Constructions of Secret-Sharing Schemes

In this section we describe some of the most interesting constructions of secret-sharing schemes.

3.1 Shamir's Threshold Secret-Sharing Scheme

In a threshold secret-sharing schemes, the authorized sets are all sets whose size is bigger than some threshold, that is, they realize the t-out-of-n access structure $\mathcal{A}_t = \{B \subseteq \{p_1, \ldots, p_n\} : |B| \geq t\}$, where $1 \leq t \leq n$ is an integer. Shamir [58] constructed a simple and elegant threshold scheme. In Shamir's scheme the domain of secrets and shares is the elements of a finite field $\mathbb{F}_q$ for some prime-power $q > n$. Let $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_q$ be n distinct non-zero elements known to all parties (e.g., if $q > n$ is a prime, then we can take $\alpha_j = j$). To share a secret $k \in \mathbb{F}_q$, the dealer chooses $t-1$ random elements $a_1, \ldots, a_{t-1}$ from $\mathbb{F}_q$ independently with uniform distribution. These random elements together with the secret define a polynomial $P(x) = k + \sum_{i=1}^{t} a_i x^i$. The share of p_j is $s_j = P(\alpha_j)$ (where P is evaluated using the arithmetic of $\mathbb{F}_q$).

The correctness and privacy of Shamir's scheme follow from the Lagrange's interpolation theorem: For every field $\mathbb{F}$, every t distinct values $x_1, \ldots, x_t$, and any t values $y_1, \ldots, y_t$, there exists a unique polynomial Q of degree at most $t-1$ over $\mathbb{F}$ such that $Q(x_j) = y_j$ for $1 \leq j \leq t$.

To see that Shamir's scheme is correct, notice that every set B of size t holds t points of the polynomial P, hence we can reconstruct it using Lagrange's interpolation, and compute $k = P(0)$. Formally, a set $B = \{p_{i_1}, \ldots, p_{i_t}\}$ computes

$$Q(x) = \sum_{\ell=1}^{t} s_{i_\ell} \prod_{1 \leq j \leq t, j \neq \ell} \frac{\alpha_{i_j} - x}{\alpha_{i_j} - \alpha_{i_\ell}}.$$

Notice that $Q(\alpha_{i_\ell}) = s_{i_\ell} = P(\alpha_{i_\ell})$ for $1 \leq \ell \leq t$. That is, P and Q are polynomial of degree at most $t-1$ that agree on t points, thus, by the uniqueness in the interpolation theorem, P and Q are equal, and, in particular, $Q(0) = P(0) = k$. Thus, the parties in B reconstruct k by computing

$$k = Q(0) = \sum_{\ell=1}^{t} s_{i_\ell} \prod_{1 \leq j \leq t, j \neq \ell} \frac{\alpha_{i_j}}{\alpha_{i_j} - \alpha_{i_\ell}}.$$

For a given set B, the reconstruction function is a linear combination of the shares, that is,

$$k = \sum_{\ell=1}^{t} \beta_\ell \cdot s_{i_\ell}, \text{ where } \beta_\ell = \prod_{1 \leq j \leq t, j \neq \ell} \frac{\alpha_{i_j}}{\alpha_{i_j} - \alpha_{i_\ell}}. \tag{7}$$

Notice that $\beta_1, \ldots, \beta_t$ depend only on the set B and not on the secret k.

On the other hand, any unauthorized set T with $t-1$ parties holds $t-1$ points of the polynomial, which together with every possible secret (a value of the polynomial in the point 0) determines a unique polynomial of degree at most $t-1$. Formally, by the interpolation theorem, for every $T = \{p_{i_1}, \ldots, p_{i_{t-1}}\}$ and every $a \in \mathbb{F}_q$, there is a unique polynomial P_a with degree at most $t-1$ such that $P_a(0) = a$ and $P_a(\alpha_{i_\ell}) = s_{i_\ell}$ for $1 \leq \ell \leq t-1$. Hence,

$$\Pr[\, \Pi(a, r)_T = \langle s_{i_\ell} \rangle_{1 \leq \ell \leq t-1} \,] = \frac{1}{q^{t-1}}.$$

Since this probability is the same for every $a \in \mathbb{F}_q$, the privacy follows.

3.2 Undirected s-t-Connectivity

Consider the access structure $\mathcal{A}_{\text{ustcon}}$, whose parties correspond to *edges* of a complete undirected graph with m vertices $v_1, \ldots, v_m$, that is, there are $n = \binom{m}{2}$ parties in the access structure, and a party is an edge (v_i, v_j), where $i < j$. A set of parties (edges) is in the access structure if the set contains a path from v_1 to v_m. Benaloh and Rudich [15] constructed a secret-sharing scheme realizing this access structure. We next describe this secret-sharing scheme. Let $k \in \{0,1\}$ be a secret. To share k, the dealer chooses $m-2$ random bits $r_2, \ldots, r_{m-1}$ independently with uniform distribution. Furthermore, the dealer sets $r_1 = k$ and $r_m = 0$. The share of a party (v_i, v_j) is $r_i \oplus r_j$.[3] To see that this scheme is correct, consider a set of parties which is a path $v_1 = v_{i_1}, v_{i_2}, \ldots, v_{i_{\ell-1}}, v_{i_\ell} = v_m$, and consider the exclusive or of the shares given to the parties (edges) of the path:

$$(r_{i_1} \oplus r_{i_2}) \oplus (r_{i_2} \oplus r_{i_3}) \oplus \cdots \oplus (r_{i_{\ell-2}} \oplus r_{i_{\ell-1}}) \oplus (r_{i_{\ell-1}} \oplus r_{i_\ell}) = r_{i_1} \oplus r_{i_\ell} = r_1 \oplus r_m = k.$$

To see that this scheme is private consider an unauthorized set, that is, a set of edges T not containing a path from v_1 to v_m. Define the set of vertices V_1 such that $v_i \in V_1$ if there exist a path in the graph (V, T) from v_1 to v_i. By definition, $v_1 \in V_1$ and $v_m \notin V_1$. Furthermore, for every $(v_i, v_j) \in T$ either both vertices v_i, v_j are in V_1 or both of them are not in V_1.

Let $\{s_{i,j}\}_{(i,j)\in T}$ be a set of shares generated for the parties in T with the secret $k = 0$, where $s_{i,j}$ is the share given to the party (v_i, v_j). We next show that the number of vectors of random bits $r_1, r_2, \ldots, r_m$ that generate $\{s_{i,j}\}_{(i,j)\in T}$ given the secret $k = 0$ is equal to the number of vectors of random bits that generate these shares given the secret $k = 1$. Fix a vector of random bits $r_1, r_2, \ldots, r_{m-1}, r_m$ that generate the shares $\{s_{i,j}\}_{(i,j)\in T}$ with the secret $k = 0$. Recall that $r_1 = k = 0$ and $r_m = 0$. Consider the random bits $r'_1, \ldots, r'_m$, where $r'_i = \overline{r_i}$ if $v_i \in V_1$ and $r'_i = r_i$ otherwise. Notice that $r'_1 = 1$ and $r'_m = 0$. Thus, these bits generate shares for the secret $k' = 1$. We claim that the random bits $r'_1, \ldots, r'_m$ generate the shares $\{s_{i,j}\}_{(i,j)\in T}$ with the secret $k = 1$. There are only two cases to consider:

- For every $(v_i, v_j) \in T$ such that $v_i, v_j \in V_1$

$$r'_i \oplus r'_j = \overline{r_i} \oplus \overline{r_j} = r_i \oplus r_j = s_{i,j}.$$

- For every $(v_i, v_j) \in T$ such that $v_i, v_j \notin V_1$

$$r'_i \oplus r'_j = r_i \oplus r_j = s_{i,j}.$$

[3] We can generalize this scheme such that the domain of secrets is any finite group H. To share a secret $k \in H$, the dealer chooses $m-2$ random elements $r_2, \ldots, r_{m-1}$ from H independently with uniform distribution, and sets $r_1 = k$ and $r_m = 0$. The share of a party (v_i, v_j), where $i < j$, is $r_j - r_i$.

To conclude, the number of vectors of random bits that generate the shares $\{s_{i,j}\}_{(i,j)\in T}$ given the secret 0 is the same as the number of vectors of random bits that generate these shares given the secret 1. This implies that the scheme is private.

3.3 Ito, Saito, and Nishizeki's Constructions [45]

Ito, Saito, and Nishizeki [45] defined secret-sharing schemes for general access structures and showed how to construct such schemes for every monotone access structure. Specifically, let $\mathcal{A}$ be any monotone access structure. The dealer shares the secret independently for each authorized set $B \in \mathcal{A}$. That is, to share a secret $k \in \{0,1\}$, the dealer does the following for every authorized set $B \in \mathcal{A}$, where $B = \{p_{i_1}, \ldots, p_{i_\ell}\}$:

- chooses $\ell - 1$ random bits $r_1, \ldots, r_{\ell-1}$,
- computes $r_\ell = k \oplus r_1 \oplus \cdots \oplus r_{\ell-1}$, and
- gives p_{i_j} the bit r_j.

We emphasize that for each set $B \in \mathcal{A}$ the random bits are chosen by the dealer independently. Clearly, each set in $\mathcal{A}$ can reconstruct the secret by computing the exclusive-or of the bits given to the set. On the other hand, each unauthorized set $T \notin \mathcal{A}$ misses at least one party from each authorized set, thus, misses at least one bit given to the authorized set. In other words, the bits held by the parties in T are uniformly distributed and independent of the secret.

To summarize, the number of bits that p_j gets is the number of authorized sets that contain p_j. A simple optimization is to share the secret k only for minimal authorized sets. Still, this scheme is highly inefficient for access structures in which the number of minimal sets is big. For example, consider the $n/2$-out-of-n access structure, that is, the access structure

$$\mathcal{A}_{n/2} = \{B \subset \{p_1, \ldots, p_n\} \,:\, |B| \geq n/2\}\,.$$

The number of bits that each party gets in the scheme of [45] is $\binom{n-1}{n/2-1} = \Theta(2^n/\sqrt{n})$. On the other hand, Shamir's scheme for this access structure gives each party a share whose size is the same as the size of the secret.

3.4 The Monotone Formulae Construction [14]

Benaloh and Leichter [14] describe a construction of secret-sharing schemes for any access structure based on monotone formulae. The construction of [14] generalizes the construction of [45] and is more efficient. However, also in this scheme for most access structures the length of the shares is exponential in the number of parties even for a one-bit secret.

The scheme of Benaloh and Leichter is recursive. It starts with schemes for simple access structures and constructs a scheme for a composition of the access structures. Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be two access structures. We assume that they have the same set of parties $\{p_1, \ldots, p_n\}$. However, it is possible that some parties are

redundant in one of the access structures, that is, there might be parties that do not belong to minimal authorized sets in one of the access structures. We define two new access structures, where $B \in \mathcal{A}_1 \vee \mathcal{A}_2$ iff $B \in \mathcal{A}_1$ or $B \in \mathcal{A}_2$, and $B \in \mathcal{A}_1 \wedge \mathcal{A}_2$ iff $B \in \mathcal{A}_1$ and $B \in \mathcal{A}_2$. We assume that for $i \in \{1,2\}$ there is a secret-sharing scheme Σ_i realizing $\mathcal{A}_i$, where the two schemes have same domain of secrets $K = \{0,\ldots,m-1\}$ for some $m \in \mathbb{N}$. Furthermore, assume that for every $1 \leq j \leq n$ the share of p_j in the scheme Σ_i is an element in $K^{a_{i,j}}$ for every $i \in \{1,2\}$, and denote $a_j = a_{1,j} + a_{2,j}$. Then there exist secret-sharing schemes realizing $\mathcal{A}_1 \vee \mathcal{A}_2$ and $\mathcal{A}_1 \wedge \mathcal{A}_2$ in which the domain of shares of p_j is K^{a_j}:

- To share a secret $k \in K$ for the access structure $\mathcal{A}_1 \vee \mathcal{A}_2$, independently share k using the scheme Σ_i (realizing $\mathcal{A}_i$) for $i \in \{1,2\}$.
- To share a secret $k \in K$ for the access structure $\mathcal{A}_1 \wedge \mathcal{A}_2$, choose $k_1 \in K$ with uniform distribution and let $k_2 = (k - k_1) \bmod m$. Next, for $i \in \{1,2\}$, independently share k_i using the scheme Σ_i (realizing $\mathcal{A}_i$). For every set $B \in \mathcal{A}_1 \wedge \mathcal{A}_2$, the parties in B can reconstruct both k_1 and k_2 and compute $k = (k_1 + k_2) \bmod m$. On the other hand, for every set $T \notin \mathcal{A}$, the parties in T do not have any information on at least one k_i, hence do not have any information on the secret k.

For example, given an access structure $\mathcal{A} = \{B_1,\ldots,B_\ell\}$, we define $\mathcal{A}_i = \{B_1,\ldots,B_i\}$. Clearly, $\mathcal{A}_i = \mathcal{A}_{i-1} \vee \{B_i\}$, and for every $1 \leq i \leq \ell$ there is a scheme realizing $\{B_i\}$ with a domain of secrets $\{0,1\}$, where each $p_j \in B$ gets a one-bit share. Applying the scheme of Benaloh and Leichter recursively, we get the scheme of Ito, Saito, and Nishizeki.

The scheme of Benaloh and Leichter can efficiently realize a much richer family of access structures than the access structures that can be efficiently realized by the scheme of Ito, Saito, and Nishizeki. To describe the access structures that can be efficiently realized by Benaloh and Leichter's scheme it is convenient to view an access structure as a function. We describe each set $A \subseteq \{p_1,\ldots,p_n\}$ by its characteristic vector (string) $v_A \in \{0,1\}^n$, where $v_A[j] = 1$ iff $p_j \in A$. With an access structure $\mathcal{A}$, we associate the function $f_{\mathcal{A}} : \{0,1\}^n \to \{0,1\}$, where $f_{\mathcal{A}}(v_B) = 1$ iff $B \in \mathcal{A}$. We say that $f_{\mathcal{A}}$ describes $\mathcal{A}$. As $\mathcal{A}$ is monotone, the function $f_{\mathcal{A}}$ is monotone. Furthermore, for two access structures $\mathcal{A}_1$ and $\mathcal{A}_2$ if $f_1 = f_{\mathcal{A}_1}$ and $f_2 = f_{\mathcal{A}_2}$, then $f_1 \vee f_2 = f_{\mathcal{A}_1 \vee \mathcal{A}_2}$ and $f_1 \wedge f_2 = f_{\mathcal{A}_1 \wedge \mathcal{A}_2}$. Using this observation, the scheme of Benaloh and Leichter can efficiently realize every access structure that can be described by a small monotone formula.[4]

Lemma 1. *Let $\mathcal{A}$ be an access structure and assume that $f_{\mathcal{A}}$ can be computed by a monotone formula in which for every $1 \leq j \leq n$, the variable x_j appears a_j times in the formula. Then, for every $m \in \mathbb{N}$, $\mathcal{A}$ can be realized with domain of secrets $\mathbb{Z}_m$ by the scheme of [14]. The resulting scheme has information ratio $\max_{1 \leq j \leq n} a_j$.*

[4] A monotone formula is a formula with OR and AND gates without negations and without negated variables. The size of such formula is the number of leaves in the tree describing the formula. A monotone formula computes a monotone function.

Any monotone Boolean function over n variables can be computed by a monotone formula. Thus, every access structure can be realized by the scheme of [14]. However, for most monotone functions, the size of the smallest monotone formula computing them is exponential in n; i.e., the information ratio of the resulting scheme is exponential in the number of the parties.

3.5 The Monotone Span Programs Construction [21,46]

All the above constructions are linear, that is, the distribution scheme is a linear mapping. More formally, in a linear secret-sharing scheme over a finite field $\mathbb{F}$, the secret is an element of the field, the random string is a vector over the field such that each coordinate of this vector is chosen independently with uniform distribution from the field, and each share is a vector over the field such that each coordinate of this vector is some fixed linear combination of the secret and the coordinates of the random string.

Example 2. Consider the scheme for $\mathcal{A}_{\text{ustcon}}$ described in Section 3.2. This scheme is linear over the field with two elements $\mathbb{F}_2$. In particular, the randomness is a vector $\langle r_2, \ldots, r_{|V|-1} \rangle$ of $|V| - 2$ random elements in $\mathbb{F}_2$, and the share of an edge (v_1, v_2), for example, is $(k + r_2) \bmod 2$, that is, this is the linear combination where the coefficient of k and r_2 are 1 and all other coefficients are zero.

To model a linear scheme, we use *monotone span programs*, which is, basically, the matrix describing the linear mapping of the linear scheme. The monotone span program also defines the access structure which the secret-sharing scheme realizes. In the rest of the paper, vectors are denoted by bold letters (e.g., $\mathbf{r}$) and, according to the context, vectors are either row vectors or column vectors (i.e., if we write $\mathbf{r}M$, then $\mathbf{r}$ is a row vector, if we write $M\mathbf{r}$, then $\mathbf{r}$ is a column vector).

Definition 4 (Monotone Span Program [46]). *A monotone span program is a triple $\mathcal{M} = (\mathbb{F}, M, \rho)$, where $\mathbb{F}$ is a field, M is an $a \times b$ matrix over $\mathbb{F}$, and $\rho : \{1, \ldots, a\} \to \{p_1, \ldots, p_n\}$ labels each row of M by a party.*[5] *The size of $\mathcal{M}$ is the number of rows of M (i.e., a). For any set $A \subseteq \{p_1, \ldots, p_n\}$, let M_A denote the sub-matrix obtained by restricting M to the rows labeled by parties in A. We say that $\mathcal{M}$ accepts B if the rows of M_B span the vector $\mathbf{e_1} = (1, 0, \ldots, 0)$. We say that $\mathcal{M}$ accepts an access structure $\mathcal{A}$ if $\mathcal{M}$ accepts a set B iff $B \in \mathcal{A}$.*

Example 3. Consider the following monotone span program $(\mathbb{F}_{17}, M, \rho)$, where

$$M = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \\ 1 & 4 & 16 \end{pmatrix}$$

[5] For simplicity, in this survey we label a row by a party p_j rather than by a variable x_j as done in [46].

and $\rho(1) = \rho(2) = p_2$, $\rho(3) = p_1$, and $\rho(4) = p_3$. Consider the sets $B = \{p_1, p_2\}$ and $T = \{p_1, p_3\}$. In this case

$$M_B = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \end{pmatrix} \text{ and } M_T = \begin{pmatrix} 1 & 3 & 9 \\ 1 & 4 & 16 \end{pmatrix}.$$

As M_B has full rank, the rows of M_B span $\mathbf{e_1}$, i.e., $(3, 14, 1)M_B = \mathbf{e_1}$ (in $\mathbb{F}_{17}$). Hence, the span program accepts $\{p_1, p_2\}$. On the other hand, the rows of M_T do not span $\mathbf{e_1}$ and the span program does not accept $\{p_1, p_3\}$. The minimal authorized sets in the access structure accepted by $\mathcal{M}$ are $\{p_1, p_2\}$ and $\{p_2, p_3\}$.

A monotone span program implies a linear secret-sharing scheme for an access structure containing all the sets accepted by the program as stated below.

Claim 2 ([21,46]). *Let $\mathcal{M} = (\mathbb{F}, M, \rho)$ be a monotone span program accepting an access structure $\mathcal{A}$, where $\mathbb{F}$ is a finite field and for every $j \in \{1, \ldots, n\}$ there are a_j rows of M labeled by p_j. Then, there is a linear secret-sharing scheme realizing $\mathcal{A}$ such that the share of party p_j is a vector in $\mathbb{F}^{a_j}$. The information ratio of the resulting scheme is $\max_{1 \le j \le n} a_j$,*

Proof. Given the monotone span program $\mathcal{M} = (\mathbb{F}, M, \rho)$, where M is an $a \times b$ matrix over $\mathbb{F}$, define a linear secret-sharing scheme as follows:

- **Input:** a secret $k \in \mathbb{F}$.
- Choose $b-1$ random elements $r_2, \ldots, r_b$ independently with uniform distribution from $\mathbb{F}$ and define $\mathbf{r} = (k, r_2, \ldots, r_b)$.
- Evaluate $(s_1, \ldots, s_a) = M\mathbf{r}$, and distribute to each player p_j the a_j entries corresponding to rows labeled by p_j.

In this linear secret-sharing scheme, every set in $\mathcal{A}$ can reconstruct the secret: Let $B \in \mathcal{A}$ and $N = M_B$, thus, the rows of N span $\mathbf{e_1}$, and there exists some vector $\mathbf{v}$ such that $\mathbf{e_1} = \mathbf{v}N$. Notice that the shares of the parties in B are $N\mathbf{r}$. The parties in B can reconstruct the secret by computing $\mathbf{v}(N\mathbf{r})$ since

$$\mathbf{v}(N\mathbf{r}) = (\mathbf{v}N)\mathbf{r} = \mathbf{e_1} \cdot \mathbf{r} = k.$$

We next prove that this scheme is private. If $T \notin \mathcal{A}$, then the rows of M_T do not span the vector $\mathbf{e_1}$, i.e., $\text{rank}(M_T) < \text{rank}\binom{M_T}{\mathbf{e_1}}$ (where $\binom{M_T}{\mathbf{e_1}}$ is the matrix containing the rows of M_T and an additional row $\mathbf{e_1}$). By simple linear algebra, $|\text{kernel}(M_T)| > \left|\text{kernel}\binom{M_T}{\mathbf{e_1}}\right|$, and there is some vector $\mathbf{w} \in \mathbb{F}^b$ such that $(M_T)\mathbf{w} = \mathbf{0}$ and $\mathbf{e_1} \cdot \mathbf{w} = 1$ (that is, $w_1 = 1$). We next prove that for every vector of shares $(s_1, \ldots, s_{|T|})$ for the parties in T, the probability that it is generated is the same for every secret $k \in \mathbb{F}$. Fix a vector $\mathbf{r} = (0, r_2, \ldots, r_b)$ such that $(M_T)\mathbf{r} = (s_1, \ldots, s_{|T|})$, that is, $\mathbf{r}$ is a vector generating shares for the secret $k = 0$. For any $k \in \mathbb{F}$ and consider the vector $\mathbf{r}' = \mathbf{r} + k\mathbf{w}$. As $r'_1 = k$, the vector $\mathbf{r}'$ generates shares for the secret k. Furthermore,

$$(M_T)\mathbf{r}' = (M_T)(\mathbf{r} + k\mathbf{w}) = (M_T)\mathbf{r} + k(M_T)\mathbf{w} = (M_T)\mathbf{r} = (s_1, \ldots, s_{|T|}).$$

That is, for every $k \in K$ the number of random strings that generate the shares $(s_1, \ldots, s_{|T|})$ when the secret is k is the same as the number of random strings that generate these shares when the secret is 0, and the scheme is private. □

Remark 2 (Historical Notes). Brickell [21] in 1989 implicitly defined monotone span programs for the case that each party labels exactly one row, and proved Claim 2. Karchmer and Wigderson [46] in 1993 explicitly defined span programs and monotone span programs. They considered them as a computational model and their motivation was proving lower bounds for modular branching programs. Karchmer and Wigderson showed that monotone span programs imply (linear) secret-sharing schemes. Beimel [3] proved that linear secret-sharing schemes imply monotone span programs. Thus, linear secret-sharing schemes are equivalent to monotone span programs, and lower bounds on the size of monotone span programs imply the same lower bounds on the information ratio of linear secret-sharing schemes.

Example 4. We next describe the linear secret-sharing for $\mathcal{A}_{\text{ustcon}}$, presented in Section 3.2, as a monotone span program. In this access structure, we consider a graph with m vertices and $n = \binom{m}{2}$ edges, each edge is a party. We construct a monotone span program over $\mathbb{F}_2$, which has $b = m - 1$ columns and $a = n$ rows. For every party (edge) (v_i, v_j), where $1 \leq i < j \leq m-1$, there is a unique row in the program labeled by this party; in this row all entries in the row are zero, except for the ith and the jth entries which are 1. Furthermore, for every party (edge) (v_i, v_m), where $1 \leq i \leq m-1$, there is a unique row in the program labeled by this party; in this row all entries in the row are zero, except for the ith entry which is 1 (this is equivalent to choosing $r_m = 0$ in Section 3.2). It can be proved that this monotone span program accepts a set of parties (edges) if and only if the set contains a path from v_1 to v_m.

To construct a secret-sharing scheme from this monotone span program, we multiply the above matrix by a vector $\mathbf{r} = (k, r_2, \ldots, r_{m-1})$ and the share of party (v_i, v_j) is the row labeled by (v_i, v_j) in the matrix multiplied by $\mathbf{r}$, that is, the share is as defined in the scheme for $\mathcal{A}_{\text{ustcon}}$ described above.

3.6 Multi-Linear Secret-Sharing Schemes [16,32]

In the schemes derived from monotone span programs, the secret is one element from the field. This can be generalized to the case where the secret is some vector over the field. Such schemes, studied by [16,32], are called multi linear and are based on the following generalization of monotone span programs.

Definition 5 (Multi-Target Monotone Span Program). *A multi-target monotone span program is a quadruple $\mathcal{M} = (\mathbb{F}, M, \rho, V)$, where $\mathbb{F}$ is a finite field, M is an $a \times b$ matrix over $\mathbb{F}$, $\rho : \{1, \ldots, a\} \to \{p_1, \ldots, p_n\}$ labels each row of M by a party, and $V = \{\mathbf{e_1}, \ldots, \mathbf{e_c}\}$ is a set of vectors in $\mathbb{F}^b$ for some $1 \leq c < b$ such that for every $A \subseteq \{p_1, \ldots, p_n\}$ either*

- *The rows of M_A span each vector in $\{\mathbf{e_1}, \ldots, \mathbf{e_c}\}$. In this case, we say that $\mathcal{M}$ accepts A, or,*

– *The rows of M_A span no non-zero vector in the linear space spanned by* $\{\mathbf{e_1}, \ldots, \mathbf{e_c}\}$.

We say that $\mathcal{M}$ accepts an access structure $\mathcal{A}$ if $\mathcal{M}$ accepts a set B iff $B \in \mathcal{A}$.

Claim 3. *Let $\mathcal{M} = (\mathbb{F}, M, \rho, V)$ be a multi-target monotone span program accepting $\mathcal{A}$, where $\mathbb{F}$ is a finite field, $|V| = c$, and for every $j \in \{1, \ldots, n\}$ there are a_j rows of M labeled by p_j. Then, there is a multi-linear secret-sharing scheme realizing $\mathcal{A}$ such that the secret is a vector in $\mathbb{F}^c$ and the share of party p_j is a vector in $\mathbb{F}^{a_j}$; in particular, the information ratio of the scheme is* $\max_{1 \le j \le n} a_j/c$.

The proof of Claim 3 is similar to the proof of Claim 2, where in this case the secret is $k_1, \ldots, k_c$, the dealer chooses $b - c$ random elements $r_{c+1}, \ldots, r_b$ in $\mathbb{F}$, uses the vector $\mathbf{r} = (k_1, \ldots, k_c, r_{c+1}, \ldots, r_b)$, and computes the shares $M\mathbf{r}$. Any multi-target monotone span program is a monotone span program; however, using it to construct a multi-linear secret-sharing scheme results in a scheme with better information ratio.

Example 5. Let $\sqcap$ be the access structure with 4 parties p_1, p_2, p_3, p_4 whose minimal authorized sets are $\{p_1, p_2\}, \{p_2, p_3\}, \{p_3, p_4\}$. It was proved by [23] that in any secret-sharing scheme realizing $\sqcap$ the information ratio is at least 1.5. We present this lower bound and prove it in Theorem 1. By definition, the information ratio of a linear scheme is integral. We next present a multi-linear secret-sharing scheme realizing $\sqcap$ with information ratio 1.5. We first describe a linear scheme whose information ratio is 2. To share a bit $k_1 \in \mathbb{F}_2$, the dealer independently chooses two random bits r_1 and r_2 with uniform distribution. The share of p_1 is r_1, the share of p_2 is $r_1 \oplus k_1$, the share of p_3 is two bits, r_1 and $r_2 \oplus k_1$, and the share of p_4 is r_2. Clearly, this scheme realizes $\sqcap$.

Notice that although p_2 and p_3 have symmetric roles in $\sqcap$, in the above scheme p_2 gets one bit and p_3 gets two bits. To construct a multi-linear scheme realizing $\sqcap$ whose information ratio is 1.5, we exploit the asymmetry of the previous scheme. To share a secret $(k_1, k_2) \in (\mathbb{F}_2)^2$, the dealer independently chooses four random bits r_1, r_2, r_3, and r_4 with uniform distribution. The scheme is described in the following table.

Share of p_1	Share of p_2	Share of p_3	Share of p_4
r_1, r_3	$r_1 \oplus k_1, r_3 \oplus k_2, r_4$	$r_1, r_2 \oplus k_1, r_4 \oplus k_2$	r_2, r_4

The secret in the above scheme is two bits and the largest shares are 3 bits, hence the information ratio of this scheme is 1.5. It is an easy exercise to write the above multi-linear scheme as a multi-target monotone span program; the matrix of this program has 10 rows and 6 columns.

The scheme in Example 5 involves two applications of linear secret-sharing schemes realizing $\sqcap$, each application with an independent secret. In particular, the multi-linear secret-sharing scheme has the same average information ratio as the linear scheme. Simonis and Ashikhmin [62] construct a multi-linear secret-sharing scheme realizing some access structure with information ratio and average information ratio 1. Furthermore, using results on representation of matroids,

they prove that any linear secret-sharing scheme realizing this access structure has average information ratio greater than 1. Thus, multi-linear secret-sharing schemes are more efficient than linear secret-sharing schemes. The maximum possible improvement in the information ratio and average information ratio of multi-linear secret-sharing schemes compared to linear secret-sharing schemes is open.

3.7 Other Constructions

There are many other constructions of secret-sharing schemes for other specific access structures, e.g., hierarchical access structures [60,21,64,66], weighted threshold access structures [11], and more complicated compositions of access structures [63,34].

4 Secret Sharing and Secure Multi-party Computation

Secret-sharing schemes are a basic building box in construction of many cryptographic protocols. In this section we demonstrate the use of secret-sharing schemes for secure multi-party computation of general functions. For simplicity we concentrate on the case that the parties are honest-but-curious, that is, the parties follow the instructions of the protocol, however, at the end of the protocol some of them might collude and try to deduce information from the messages they got. The protocols that we describe are secure against an all-powerful adversary, that is, they supply information-theoretic security.

Definition 6 (Secure Computation in the Honest-but-Curious Model (Informal)). *Let $\mathbb{F}$ be a finite field. Assume there are n parties $p_1, \ldots, p_n$, and at most t of them are corrupted, where $t < n$. Each party p_j holds a private input $x_j \in \mathbb{F}$. The parties want to compute some function $f(x_1, \ldots, x_n)$ by exchanging messages on private channels according to some protocol $\mathcal{P}$. We have two requirements:*

Correctness. *At the end of the protocol each party outputs $f(x_1, \ldots, x_n)$.*
Privacy. *Every coalition T of at most t parties cannot learn any information not implied by the inputs $\{x_j\}_{p_j \in T}$ and the output of the function.*

We will first show a homomorphic property of Shamir's secret-sharing scheme. Using this property, we show how to use secret sharing to construct a protocol for securely computing the sum of secret inputs. Then, we will show how to securely compute the product of inputs. Combining these protocols we get an efficient protocol for computing any function which can be computed by a small arithmetic circuit. Such protocols with information-theoretic security were first presented in [13,24]. The exact protocol we present here is from [38].

Claim 4. *Let $k_1, k_2 \in \mathbb{F}$ be two secrets. For $i \in \{1, 2\}$, let $s_{i,1}, \ldots, s_{i,n}$ be a sharing of k_i using Shamir's $(t+1)$-out-of-n scheme (see Section 3.1). Then, $s_{1,1} + s_{2,1}, \ldots, s_{1,n} + s_{2,n}$ are shares of the secret $k_1 + k_2$ in Shamir's $(t+1)$-out-of-n scheme. Similarly, $s_{1,1} \cdot s_{2,1}, \ldots, s_{1,n} \cdot s_{2,n}$ are shares of the secret $k_1 \cdot k_2$ in Shamir's $(2t+1)$-out-of-n scheme.*

Proof. Let Q_1 and Q_2 be the polynomial of degree at most t generating the shares $s_{1,1}, \ldots, s_{1,n}$ and $s_{2,1}, \ldots, s_{2,n}$ respectively, that is $Q_i(0) = k_i$ and $Q_i(\alpha_j) = s_{i,j}$ for $i \in \{1,2\}$ and $1 \leq j \leq n$ (where $\alpha_1, \ldots, \alpha_n$ are defined in Section 3.1). Define $Q(x) = Q_1(x) + Q_2(x)$. This is a polynomial of degree at most t such that $Q(0) = Q_1(0) + Q_2(0) = k_1 + k_2$ and $Q(\alpha_j) = s_{1,j} + s_{2,j}$, that is, this is a polynomial generating the shares $s_{1,1} + s_{2,1}, \ldots, s_{1,n} + s_{2,n}$ given the secret $k_1 + k_2$.

Similarly, let $R(x) = Q_1(x) \cdot Q_2(x)$. This is a polynomial degree at most $2t$ generating the shares $s_{1,1} \cdot s_{2,1}, \ldots, s_{1,n} \cdot s_{2,n}$ given the secret $k_1 \cdot k_2$.[6] □

4.1 Computing the Sum of Two Shared Numbers

Assume that two secrets x_1 and x_2 are shared using Shamir's $(t+1)$-out-of-n secret-sharing scheme. Using Claim 4, each party can compute a share of the sum of the secrets without any communication.

Input of party p_j. Shares $s_{1,j}$ and $s_{2,j}$ of the secrets x_1 and x_2 respectively.
Computation step: Each party p_j computes $s_j = s_{1,j} + s_{2,j}$.

4.2 Computing the Product of Two Shared Numbers

Assume that two secrets x_1 and x_2 are shared using Shamir's $(t+1)$-out-of-n secret-sharing scheme. Using Claim 4, the parties can compute shares of the product $x_1 \cdot x_2$ in a $(2t+1)$-out-of-n secret-sharing scheme. We show that, by using interaction, the parties can compute shares of the product $x_1 \cdot x_2$ in Shamir's $(t+1)$-out-of-n secret-sharing scheme (without learning the product itself). In this case, we assume that there are t corrupt parties, where $n = 2t+1$ (that is, there is a majority of honest parties).

Input of party p_j. Shares $s_{1,j}$ and $s_{2,j}$ of the secrets x_1 and x_2 respectively in Shamir's $(t+1)$-out-of-n secret-sharing scheme.

Step I. Each party p_j computes $s_j = s_{1,j} \cdot s_{2,j}$ and shares s_j using Shamir's $(t+1)$-out-of-n secret-sharing scheme. Denote the resulting shares by $q_{j,1}, \ldots, q_{j,n}$. Party p_j sends $q_{j,\ell}$ to p_ℓ.

Step II. Let $\beta_1, \ldots, \beta_\ell$ be the constants defined in (7) for the reconstruction of the secret in Shamir's $(2t+1)$-out-of-n scheme. Each party p_ℓ computes $u_\ell = \sum_{j=1}^{n} \beta_j q_{j,\ell}$.

We next explain why this protocol is correct. By Claim 4, $s_1, \ldots, s_n$ are shares of $x_1 \cdot x_2$ in a Shamir's $(2t+1)$-out-of-n scheme. Thus, by (7), $x_1 \cdot x_2 = \sum_{j=1}^{n} \beta_j s_j$. As $q_{j,1}, \ldots, q_{j,n}$ are shares in Shamir's $(t+1)$-out-of-n scheme of the secret s_j, Claim 4 implies that $u_1, \ldots, u_\ell$ are shares of $x_1 \cdot x_2$.

[6] While $Q(x)$ is a uniformly distributed polynomial such that $Q(0) = k_1 + k_2$, the polynomial $R(x)$ is *not* uniformly distributed (that is, $R(x)$ is product of two polynomials of degree t). For the protocols we present, this does not cause any problems.

4.3 Computing an Arithmetic Circuit

Using the above protocols, we show how to securely compute any function represented by an arithmetic circuit assuming that $n = 2t+1$. An arithmetic circuit over $\mathbb{F}$ with n inputs is an acyclic graph where:

- There is a unique node with out-degree 0. This node is called the output node.
- There are n nodes with in-degree 0, called input gates. For each i, where $1 \leq i \leq n$, there is a node labeled by the variable x_i.[7]
- Each internal node is labeled either by $\times$, called a multiplication gate, or by $+$, called an addition gate. Each internal node has in-degree two.

The function computed by an arithmetic circuit over a field $\mathbb{F}$ is defined in the natural way, where the arithmetic is done over $\mathbb{F}$. The complexity of computing the function is proportional to the number of gates in the circuit. We next show a secure protocol for evaluating the function computed by an arithmetic circuit, where each party p_j holds x_j. The number of rounds in this protocol is linear in the number of gates. More formally, let $G_1, G_2, \ldots, G_\ell$ be the gates of a circuit sorted according to some topological order (that is, if there exists an edge from G_j to G_i, then $i > j$). Assume that, for $1 \leq i \leq n$, the gate G_i is labeled by x_i.

The protocol for computing the arithmetic circuit keeps intermediate values as shares of a $(t+1)$-secret-sharing scheme. In the beginning of the protocol, each party shares its input. Thereafter, the protocol proceeds in phases, where in the beginning of phase i the parties hold shares of a $(t+1)$-out-of-n secret-sharing scheme of the two inputs of gate G_i, and in the end of phase i the parties hold shares of a $(t+1)$-out-of-n secret-sharing scheme of the output of the gate G_i. At the end of the protocol, the output is reconstructed from the shares.

Input of party p_j. An element $x_j \in \mathbb{F}$.

Initialization. Each party p_i shares x_i using Shamir's $(t+1)$-out-of-n secret-sharing scheme. Denote the resulting shares by $q_{i,1}, \ldots, q_{i,n}$. Party p_i sends $q_{i,j}$ to p_j.

Computation stages. For $i = n+1$ to ℓ compute shares of the output of gate G_i as follows:

- Assume that the incoming edges into gate G_i are from gates G_j and G_k, where $j, k < i$ and the parties holds shares $q_{j,1}, \ldots, q_{j,n}$ and $q_{k,1}, \ldots, q_{k,n}$ of the outputs of these gates.
- If G_i is an addition gate, each party p_m locally computes $q_{i,m} = q_{j,m} + q_{k,m}$ as the share of the output of gate G_i.
- If G_i is a multiplication gate, the parties use the one-round protocol described in Section 4.2 to compute shares of the product of the outputs of gates G_j and G_k.

[7] There can be additional nodes with in-degree 0 labeled by constants. For simplicity, we ignore such nodes.

Reconstruction. Each party p_m sends its share $q_{\ell,m}$ to p_1. Party p_1 reconstructs a secret s from the shares $q_{\ell,1}, \ldots, q_{\ell,t+1}$ using the reconstruction procedure of Shamir's $(t+1)$-out-of-n secret-sharing scheme, and sends s to all parties, which output this value.

By the correctness of the addition and multiplication protocols, at the end of phase i, the parties hold shares of the output of gate G_i. Thus, at the end of the protocol they hold shares of the output of the circuit, and s is the correct value for the output of the protocol. On the other hand, in each stage any coalition of at most t parties sees at most t shares of a $(t+1)$-out-of-n secret-sharing scheme, thus, the set does not learn information not implied by the inputs of the set and the sum.

4.4 Extensions to Other Models

The protocol we described above assumes that the corrupted parties are honest-but-curious. A more realistic assumption is that the parties can deviate from the protocol and send any messages that might help them. Such parties are called malicious. For example, in the multiplication protocol, a party that should share s_j can send shares that are not consistent with any secret. Furthermore, in the reconstruction step in the arithmetic circuit protocol, a party can send a "wrong" share. To cope with malicious behavior, the notion of *verifiable secret sharing* was introduced by Chor et al. [25]. Such schemes were constructed under various assumptions, see [38] for a partial list of such constructions. We will not elaborate on verifiable secret sharing in this survey.

In the definition of secure computation we assumed that there is a parameter t, and an adversary can control any coalition of size at most t. This assumes that all parties are as likely to be corrupted. Hirt and Maurer [42] considered a more general scenario in which there is an access structure, and the adversary can control any set of parties not in the access structure. That is, they require that any set not in the access structure cannot learn information not implied by the inputs of the parties in the set and the output of the function. Similarly to the requirement that $2t < n$ in the protocol we described above, secure computation against honest-but-curious parties is possible for general functions iff the union of every two sets not in the access structure does not cover the entire set of parties [42]. For every such access structure $\mathcal{A}$, Cramer et al. [28] showed that using every linear secret-sharing scheme realizing $\mathcal{A}$, one can construct a protocol for computing any arithmetic circuit such that any set not in the access structure cannot learn any information; the complexity of the protocol is linear in the size of the circuit. Their protocol is similar to the protocol we described above, where for addition gates every party does local computation. Multiplication is also similar, however, the choice of the constants $\beta_1, \ldots, \beta_n$ is more involved. The protocol of Cramer et al. [28] shows the need for general secret-sharing schemes.

5 Lower Bounds on the Size of the Shares

The best known constructions of secret-sharing schemes for general access structures (e.g., [45,14,21,46,16,32]) have information ratio $2^{O(n)}$, where n is the number of parties in the access structure. As discussed in the introduction, we conjecture that this is the best possible. Lower bounds for secret-sharing schemes have been proved in, e.g., [47,23,19,33,29,30,18]. However, these lower bounds are far from the exponential upper bounds. The best lower bound was proved by Csirmaz [29,30], who proved that for every n there exists an n-party access structure such that every secret-sharing scheme realizing it has information ratio $\Omega(n/\log n)$. In Sections 5.2 – 5.3, we review this proof. For linear secret-sharing schemes the situation is much better – for every n there exist access structures with n parties such that every linear secret-sharing scheme realizing them has super-polynomial, i.e., $n^{\Omega(\log n)}$, information ratio [5,2,36,37]. In Section 5.5, we present the lower bound proof of [37].

5.1 A Simple Lower Bound

Karnin et al. [47] have showed that for each non-redundant party p_j (that is, a party that appears in at least one minimal authorized set) $H(S_j) \geq H(S)$, which implies that the size of the share of the party is at least the size of the secret. We next give a direct proof of the latter result.

Lemma 2. *Let p_j be a non-redundant party in $\mathcal{A}$ and let Σ be any secret-sharing scheme realizing $\mathcal{A}$, where K and K_j are the domains of secrets and of the shares of p_j respectively. Then, $|K_j| \geq |K|$.*

Proof. Let B be a minimal authorized set in $\mathcal{A}$ containing p_j, that is $B \in \mathcal{A}$ and $B' \stackrel{\text{def}}{=} B \setminus \{p_j\} \notin \mathcal{A}$. Assume that there is a secret-sharing-scheme realizing $\mathcal{A}$ in which $|K_j| < |K|$. Fix any vector of shares $\{s_i\}_{p_i \in B'}$ for the parties of B' that has positive probability (given some secret $k_0 \in K$). By the privacy property, this vector of shares should have positive probability given any secret $k \in K$. That is, for every $k \in K$, there is a share $s^k \in K_j$ such that $\{s_i\}_{p_i \in B'}$ together with s^k have positive probability given the secret k. Since $|K_j| < |K|$, there are secrets $k_1, k_2 \in K$ such that $k_1 \neq k_2$ and $s^{k_1} = s^{k_2}$. Thus, the authorized set B holding the shares $\{s_i\}_{p_i \in B'}$ and s^{k_1} errs in the reconstruction for at least one of the secrets k_1 and k_2, contradicting the correctness of the scheme. □

5.2 Stronger Lower Bounds

Starting from the works of Karnin et al. [47] and Capocelli et al. [23], the entropy was used to prove lower bounds on the share size in secret-sharing schemes [19,33,29,30]. In other words, to prove lower bounds on the information ratio of secret-sharing schemes, we use the alternative definition of secret sharing via the entropy function, Definition 3.

Towards proving lower bounds, we use properties of the entropy function as well as the correctness and privacy of secret-sharing schemes. This is summarized in Claim 5. To simplify notations, in the sequel we denote $H(S_A)$ by $H(A)$ for any set of parties $A \subseteq \{p_1, \ldots, p_n\}$. Furthermore, we denote $H(S_A S)$ by $H(AS)$. In the lower bounds proof, we assume uniform distribution on the secrets, that is, $H(S) = \log |K|$. As proved in Claim 1, this assumption is without loss of generality. By the properties of the entropy function, for every j, $H(\{p_j\}) \leq \log |K_j|$, thus, the information ratio of the scheme, that is, $\max_{1 \leq j \leq n} \log |K_j| / \log |K|$, is at least $\max_{1 \leq j \leq n} H(\{p_j\})/H(S)$.

Claim 5. *Let $A, B \subseteq \{p_1, \ldots, p_n\}$ and Σ be a secret-sharing scheme realizing an access structure $\mathcal{A}$. The following 4 properties hold:*

Monotonicity.	*If $A \subset B$, then $H(B) \geq H(A) \geq H(\emptyset) = 0$.*
Submodularity.	$H(A) + H(B) \geq H(A \cup B) + H(A \cap B)$.
Strong Monotonicity.	*If $A \notin \mathcal{A}$, $B \in \mathcal{A}$, and $A \subset B$, then* $H(B) \geq H(A) + H(S)$.
Strong Submodularity.	*If $A, B \in \mathcal{A}$ and $A \cap B \notin \mathcal{A}$, then* $H(A) + H(B) \geq H(A \cup B) + H(A \cap B) + H(S)$.

Proof. The monotonicity and submodularity are true for any random variables (where the submodularity follows from the fact that the conditional mutual information is non-negative). For the strong monotonicity observe that by the correctness, monotonicity, and privacy, $H(B) = H(BS) \geq H(AS) = H(A) + H(S)$. For the strong submodularity, note that if $A, B \in \mathcal{A}$ and $A \cap B \notin \mathcal{A}$, then $H(AS) = H(A)$, $H(BS) = H(B)$, $H((A \cup B)S) = H(A \cup B)$, and $H((A \cap B)S) = H(A \cap B) + H(S)$. Thus, $H(A) + H(B) = H(AS) + H(BS) \geq H((A \cup B)S) + H((A \cap B)S) = H(A \cup B) + H(A \cap B) + H(S)$. □

To give an example of using Claim 5, we present the lower bound of [23] for the access structure $\sqcap$ (defined in Example 5).

Theorem 1 ([23]). *The information ratio of every secret-sharing scheme realizing $\sqcap$ is at least* 1.5.

Proof. Let Σ be any secret-sharing scheme realizing $\sqcap$. By Claim 5,

$$H(\{p_1, p_2\}) + H(\{p_2, p_3\}) \geq H(\{p_1, p_2, p_3\}) + H(\{p_2\}) + H(S), \tag{8}$$

$$H(\{p_1, p_3, p_4\}) \geq H(\{p_1, p_4\}) + H(S), \tag{9}$$

$$H(\{p_1, p_2, p_3\}) \geq H(\{p_1, p_3\}) + H(S), \tag{10}$$

$$H(\{p_1, p_3\}) + H(\{p_1, p_4\}) \geq H(\{p_1, p_3, p_4\}) + H(\{p_1\}), \tag{11}$$

$$H(\{p_1\}) + H(\{p_2\}) \geq H(\{p_1, p_2\}), \tag{12}$$

$$H(\{p_2\}) + H(\{p_3\}) \geq H(\{p_2, p_3\}). \tag{13}$$

In the above, (8) follows from strong submodularity, (9) and (10) follow from strong monotonicity, and (11), (12), and (13) follow from submodularity.

Summing all these inequalities, we get $H(\{p_2\}) + H(\{p_3\}) \geq 3H(S)$, and the information ratio of the scheme is at least

$$\max\{H(\{p_2\}), H(\{p_3\})\} / H(S) \geq 1.5.$$

□

5.3 Csirmaz's Lower Bound

We next present Csirmaz's lower bound on the information ratio. We first define, for every $n \in \mathbb{N}$, an access structure $\mathcal{A}_n$ by specifying its minimal sets. Let k be the largest integer such that $2^k + k - 1 \leq n$. Let $B = \{p_1, \ldots, p_{2^k-1}\}$ and define $B_0 = \emptyset$ and $B_i = \{p_1, \ldots, p_i\}$ for $1 \leq i \leq 2^k - 1$. Furthermore, let $A = \{p_{2^k}, \ldots, p_{2^k+k-1}\}$ (that is, $|A| = k$), and $A = A_0, A_1, \ldots, A_{2^k-1} = \emptyset$ be all the subsets of A such that if $i < i'$, then $A_i \not\subseteq A_{i'}$. Finally, define $U_i = A_i \cup B_i$ for $0 \leq i \leq 2^k - 1$. The minimal sets of $\mathcal{A}_n$ are $U_0, U_1, \ldots, U_{2^k-1}$.

Lemma 3. *For every* $0 \leq i \leq 2^k - 2$

$$H(B_i \cup A) - H(B_i) \geq H(B_{i+1} \cup A) - H(B_{i+1}) + H(S). \tag{14}$$

Proof. On the one hand, $U_i \subseteq B_i \cup A \in \mathcal{A}_n$, and $U_{i+1} = B_{i+1} \cup A_{i+1} \in \mathcal{A}_n$. On the other hand, $B_i \cup A_{i+1} \notin \mathcal{A}_n$, since $B_i \cup A_{i+1}$ does not contain any minimal authorized set U_j:

Case I: $j > i$**.** $p_{i+1} \in U_j = B_j \cup A_j$, while $p_{i+1} \notin B_i \cup A_{i+1}$,
Case I: $j \leq i$**.** $A_j \subseteq U_j$, while $A_j \not\subseteq A_{i+1}$.

Thus, by the strong submodularity,

$$H(B_i \cup A) + H(B_{i+1} \cup A_{i+1}) \geq H(B_{i+1} \cup A) + H(B_i \cup A_{i+1}) + H(S).$$

Furthermore, by submodularity,

$$H(B_i \cup A_{i+1}) + H(B_{i+1}) \geq H(B_{i+1} \cup A_{i+1}) + H(B_i).$$

Summing the last two inequalities results in (14). □

Theorem 2 ([29]). *For every* n *there exists an* n*-party access structure* $\mathcal{A}_n$ *such that every secret-sharing scheme realizing it has information ratio* $\Omega(n/\log n)$.

Proof. Summing (14) for every $0 \leq i \leq 2^k - 2$ we get that

$$H(B_0 \cup A) - H(B_0) \geq H(B_{2^k-1} \cup A) - H(B_{2^k-1}) + (2^k - 1)H(S). \tag{15}$$

By monotonicity, $H(B_{2^k-1} \cup A) - H(B_{2^k-1}) \geq 0$. Furthermore, by submodularity, $\sum_{p_j \in A} H(\{p_j\}) \geq H(A)$ and $H(B_0) + H(A) \geq H(B_0 \cup A)$. Thus,

$$\begin{aligned}
\sum_{p_j \in A} H(\{p_j\}) &\geq H(A) \\
&\geq H(B_0 \cup A) - H(B_0) \\
&\geq H(B_{2^k-1} \cup A) - H(B_{2^k-1}) + (2^k - 1)H(S) \\
&= \Omega(n) \cdot H(S).
\end{aligned}$$

This implies that $H(\{p_j\}) = \Omega(n/\log n)H(S)$ for at least one party p_j, and the information ratio of the scheme is $\Omega(n/\log n)$. □

We next show how to strengthen Theorem 2 and show that there exists an access structure in which the shares of many parties have to be long.

Theorem 3 ([30]). *For every n there exists an n-party access structure $\mathcal{A}'_n$ such that every secret-sharing scheme realizing it has* average *information ratio $\Omega(n/\log n)$.*

Proof. In the proof of Theorem 2 we constructed an access structure in which there is a small set A of size $O(\log n)$ such that the sum of the entropies of the shares given to the parties in the set is $\Omega(n)H(S)$. We next construct a similar access structure which has many copies of A and one copy of B. Let k be the largest integer such that $2^k \leq n/2$. Let $B = \{p_1, \ldots, p_{2^k-1}\}$ and define $B_0 = \emptyset$ and $B_i = \{p_1, \ldots, p_i\}$ for $1 \leq i \leq 2^k - 1$. Furthermore, let $A^\ell = \{p_{2^k+\ell k}, \ldots, p_{2^k+(\ell+1)k-1}\}$ for $0 \leq \ell \leq \lfloor n/2k \rfloor - 1$, and $A^\ell = A^\ell_0, A^\ell_1, \ldots, A^\ell_{2^k-1} = \emptyset$ be all the subsets of A^ℓ such that if $i < i'$, then $A^\ell_i \not\subseteq A^\ell_{i'}$. Finally, the minimal sets of $\mathcal{A}'_n$ are $U^\ell_i \stackrel{\text{def}}{=} A^\ell_i \cup B_i$ for $0 \leq i \leq 2^k - 1$ and $0 \leq \ell \leq \lfloor n/2k \rfloor - 1$.

For every ℓ, the access structure $\mathcal{A}'_n$ restricted to the parties in $B \cup A^\ell$ is isomorphic to the access structure $\mathcal{A}_{n'}$ (where $n' > n/2$). Thus, by (16),

$$\sum_{p_j \in A^\ell} H(\{p_j\}) \geq (2^k - 1)H(S) = \Omega(n)H(S).$$

As the sets A^ℓ are disjoint,

$$\begin{aligned}\sum_{j=1}^{n} H(\{p_j\}) &> \sum_{\ell=0}^{\lfloor n/2k \rfloor - 1} \sum_{p_j \in A^\ell} H(\{p_j\}) \geq (n/(2k) - 1)(2^k - 1)H(S) \\ &= \Omega(n^2/\log n)H(S).\end{aligned}$$

Thus, the average information ratio of $\mathcal{A}'_n$ is $\Omega(n/\log n)$. □

5.4 Limitations of Known Techniques for Lower Bounds

Basically, all known lower bounds for the size of shares in secret-sharing schemes are implied by Claim 5. In other words, they only use the so-called Shannon information inequalities (i.e., the fact that the conditional mutual information is non-negative). Csirmaz [29] in 1994 proved that such proofs cannot prove a lower bound of $\omega(n)$ on the information ratio. That is, Csirmaz's lower bound is nearly optimal (up to a factor $\log n$) using only Shannon inequalities. In 1998, new information inequalities were discovered by Zhang and Yeung [71]. Other information inequalities were discovered since, see, e.g. [70]. In particular, there are infinitely many independent information inequalities in 4 variables [50]. Such inequalities were used in [7,51] to prove lower bounds for secret-sharing schemes. Beimel and Orlov [8] proved that all information information inequalities with

4 or 5 variables and all known information inequalities in more than 5 variables cannot prove a lower bound of $\omega(n)$ on the information ratio of secret-sharing schemes. Thus, new information inequalities with more than 5 variables should be found if we want to improve the lower bounds.

5.5 Lower Bounds for Linear Secret Sharing

For linear secret-sharing schemes we can prove much stronger lower bounds than for general secret-sharing schemes. Recall that linear secret-sharing schemes are equivalent to monotone span programs and we first state the results using monotone span programs. Lower bounds for monotone span programs were presented in [5,2,36,37,10]; the best known lower bound is $n^{\Omega(\log n)}$ as proved in [36]. We present here an alternative proof of [37]. We start with a simple observation.

Observation 1. Let $\mathcal{A}$ be a (monotone) access structure. Let $B \in \mathcal{A}$ and $C \subseteq \{p_1, \ldots, p_n\}$ such that $\{p_1, \ldots, p_n\} \setminus C \notin \mathcal{A}$. Then, $B \cap C \neq \emptyset$.

The observation follows from the fact that if $B \cap C = \emptyset$, then $B \subseteq \{p_1, \ldots, p_n\} \setminus C$, contradicting the fact that $B \in \mathcal{A}$ and $\{p_1, \ldots, p_n\} \setminus C \notin \mathcal{A}$.

To prove the lower bound, Gál and Pudlák [37] choose a subset of the unauthorized sets that satisfies some properties, they use this subset to construct a matrix over $\mathbb{F}$, and prove that the rank of the matrix over $\mathbb{F}$ is a lower bound on the size of every monotone span program realizing $\mathcal{A}$.

Let $\mathcal{B} = \{B_1, \ldots, B_\ell\}$ be the collection of minimal authorized sets in $\mathcal{A}$ and $\mathcal{C} = \{(C_{1,0}, C_{1,1}), (C_{2,0}, C_{2,1}), \ldots, (C_{t,0}, C_{t,1})\}$ be a collection of pairs of sets of parties such that $\{p_1, \ldots, p_n\} \setminus (C_{j,0} \cup C_{j,1}) \notin \mathcal{A}$ for every $1 \leq j \leq t$. By Observation 1, $B_i \cap (C_{j,0} \cup C_{j,1}) \neq \emptyset$ for every i, j, that is, at least one of the following conditions hold: (1) $B_i \cap C_{j,0} \neq \emptyset$, (2) $B_i \cap C_{j,1} \neq \emptyset$. To prove the lower bound, Gál and Pudlák use a collection $\mathcal{C}$ such that, for every i, j, *exactly* one of the above conditions hold.

Definition 7. *We say that a collection $\mathcal{C}$ satisfies the* unique intersection *property for $\mathcal{A}$ if*

1. *For every $1 \leq j \leq t$, $\{p_1, \ldots, p_n\} \setminus (C_{j,0} \cup C_{j,1}) \notin \mathcal{A}$.*
2. *For every $1 \leq i \leq \ell$ and every $1 \leq j \leq t$, exactly one of the following conditions hold (1) $B_i \cap C_{j,0} \neq \emptyset$, (2) $B_i \cap C_{j,1} \neq \emptyset$.*

Example 6. Consider the access structure with ten parties $\{p_1, \ldots, p_{10}\}$ and six minimal authorized sets $\{p_1, p_2, p_5\}$, $\{p_1, p_3, p_6\}$, $\{p_1, p_4, p_7\}$, $\{p_2, p_3, p_8\}$, $\{p_2, p_4, p_9\}$, and $\{p_3, p_4, p_{10}\}$. We next define $\mathcal{C}$ satisfying the unique intersection property for $\mathcal{A}$, where $\mathcal{C}$ is $(\{p_1, p_2\}, \{p_{10}\})$, $(\{p_1, p_3\}, \{p_9\})$, $(\{p_1, p_4\}, \{p_8\})$, $(\{p_2, p_3\}, \{p_7\})$, $(\{p_2, p_4\}, \{p_6\})$, and $(\{p_3, p_5\}, \{p_1\})$.

It can be seen that $\mathcal{C}$ satisfies (1). For example, the set $T = \{p_1, \ldots, p_{10}\} \setminus (\{p_1, p_2\} \cup \{p_{10}\}) = \{p_3, p_4, \ldots, p_9\}$ is unauthorized since the only minimal authorized set that contains $\{p_3, p_4\}$ is $\{p_3, p_4, p_{10}\}$ and $p_{10} \notin T$. Furthermore, $\mathcal{C}$ satisfies (2). Consider, e.g., $\{p_1, p_3, p_6\} \in \mathcal{B}$ and $(\{p_1, p_2\}, \{p_{10}\}) \in \mathcal{C}$. In this case $\{p_1, p_3, p_6\} \cap \{p_1, p_2\} \neq \emptyset$ while $\{p_1, p_3, p_6\} \cap \{p_{10}\} = \emptyset$.

Theorem 4 ([37]). *Let $\mathcal{C}$ be a collection satisfying the unique intersection property for $\mathcal{A}$ and define an $\ell \times t$ matrix D, where $D_{i,j} = 0$ if $B_i \cap C_{j,0} \neq \emptyset$ and $D_{i,j} = 1$ if $B_i \cap C_{j,1} \neq \emptyset$. Then, the size of every monotone span program over $\mathbb{F}$ accepting $\mathcal{A}$ is at least* $\text{rank}_{\mathbb{F}}(D)$.

Example 7. The matrix D defined for the set $\mathcal{C}$ of Example 6 is the full rank matrix described below:

$$D = \begin{pmatrix} 0\,0\,0\,0\,0\,1 \\ 0\,0\,0\,0\,1\,0 \\ 0\,0\,0\,1\,0\,0 \\ 0\,0\,1\,0\,0\,0 \\ 0\,1\,0\,0\,0\,0 \\ 1\,0\,0\,0\,0\,0 \end{pmatrix}.$$

Proof (Proof of Theorem 4). Let $\mathcal{M} = (\mathbb{F}, M, \rho)$ be a monotone span program accepting $\mathcal{A}$, and denote the size of $\mathcal{M}$ (i.e., the number of rows in M) by m. We will construct two matrices L and R, where L has m columns and R has m rows such that $D = LR$. Thus, $\text{rank}_{\mathbb{F}}(D) \leq \text{rank}_{\mathbb{F}}(L) \leq m$.

Fix any i such that $1 \leq i \leq \ell$. Since $B_i \in \mathcal{A}$, the rows in M labeled by the parties in B_i span the vector $\mathbf{e_1}$, that is, there exists a vector $\mathbf{v_i}$ such that $\mathbf{v_i}M = \mathbf{e_1}$ and the non-zero coordinates of $\mathbf{v_i}$ are only in rows labeled by B_i (where the dth coordinate of $\mathbf{v_i}$ is labeled by $\rho(d)$).

Fix any j such that $1 \leq j \leq t$, and let $T_j = \{p_1, \ldots, p_n\} \setminus (C_{j,0} \cup C_{j,1})$. Since $T_j \notin \mathcal{A}$, the rows in M labeled by the parties in T_j do not span the vector $\mathbf{e_1}$. As explained in Section 3.4, there exists a vector $\mathbf{w_j}$ such that $M_{T_j}\mathbf{w_j} = \mathbf{0}$ and $\mathbf{e_1} \cdot \mathbf{w_j} = 1$. Let $\mathbf{y_j} \stackrel{\text{def}}{=} M\mathbf{w_j}$. Note that all coordinates in $\mathbf{y_j}$ labeled by the parties in T_j are zero. Furthermore, for every i, j,

$$\mathbf{v_i}\mathbf{y_j} = \mathbf{v_i}(M\mathbf{w_j}) = (\mathbf{v_i}M)\mathbf{w_j} = \mathbf{e_1} \cdot \mathbf{w_j} = 1.$$

We next modify the vectors $\mathbf{y_1}, \ldots, \mathbf{y_t}$ to vectors $\mathbf{z_1}, \ldots, \mathbf{z_t}$ such that $\mathbf{v_i}\mathbf{z_j} = D_{i,j}$ for every i, j. Let $\mathbf{z_j}$ be the column vector achieved from $\mathbf{y_j}$ by replacing all coordinates in $\mathbf{y_j}$ labeled by parties in $C_{j,0}$ with 0. Thus, only coordinates in $\mathbf{z_j}$ labeled by parties in $C_{j,1}$ can be non-zero. Hence,

- If $B_i \cap C_{j,0} \neq \emptyset$, then $D_{i,j} = 0$ and $\mathbf{v_i}$ and $\mathbf{z_j}$ do not share non-zero coordinates, thus, $\mathbf{v_i} \cdot \mathbf{z_j} = 0 = D_{i,j}$.
- If $B_i \cap C_{j,1} \neq \emptyset$, then $D_{i,j} = 1$ and all coordinates in $\mathbf{v_i}$ labeled by $C_{j,0}$ are zero, thus, $\mathbf{v_i} \cdot \mathbf{z_j} = \mathbf{v_i} \cdot \mathbf{y_j} = 1 = D_{i,j}$.

Define a matrix L, where the ith row in L is $\mathbf{v_i}$, and a matrix R, where the jth column of R is $\mathbf{z_j}$. We claim that $D = LR$ since $D_{i,j} = \mathbf{v_i} \cdot \mathbf{z_j}$. As L has m columns, $\text{rank}_{\mathbb{F}}(D) = \text{rank}_{\mathbb{F}}(LR) \leq \text{rank}_{\mathbb{F}}(L) \leq m$. In other words, the rank of D is at most the size of smallest monotone span program accepting $\mathcal{A}$. □

We next present a construction of an access structure for which we can prove an $n^{\Omega(\log n)}$ lower bound using Theorem 4. A bipartite graph $G = (U, V, E)$ has the

isolated neighbor property for t if for every two disjoint sets $A_1, A_2 \subset U$ such that $|A_1| = |A_2| = t$, there exists a vertex $v \in V$ such that $(u_1, v) \in E$ for every $u_1 \in A_1$ and $(u_2, v) \notin E$ for every $u_2 \in A_2$, that is, v is a neighbor of every vertex in A_1 and is isolated from every vertex in A_2.

For a set $A \subset U$ define $N(A) \stackrel{\text{def}}{=} \{v : \forall_{u \in A}(u, v) \in E\}$, that is, a vertex is in $N(A)$ if it is a neighbor of all vertices in A. Let $G = (U, V, E)$ be a bipartite graph satisfying the isolated neighbor property for t, where the vertices of the graph are parties, i.e., $U \cup V = \{p_1, \ldots, p_n\}$. We define an access structure $\mathcal{N}_G$ with $|U|+|V|$ parties whose minimal authorized sets are the sets $A \cup N(A)$ where $A \subset U$ and $|A| = t$.

Example 8. Consider the graph described in Figure 1. This is a trivial graph satisfying the isolated neighbor property for $t = 2$. For example, consider the disjoint sets $\{p_1, p_2\}$ and $\{p_3, p_4\}$; vertex p_5 is a neighbor of all the vertices in the first set while it is not a neighbor of any vertex in the second set. The access structure $\mathcal{N}_G$ defined for this graph is the access structure defined in Example 6.

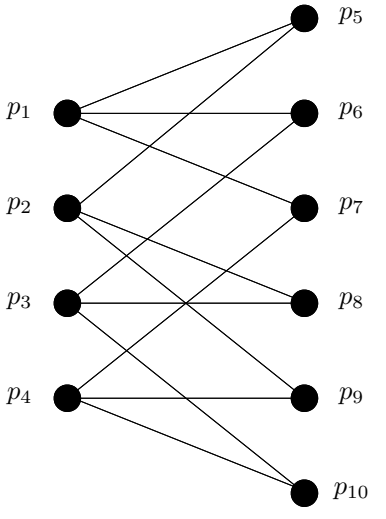


Fig. 1. An example of a graph satisfying the isolated neighbor property for $t = 2$

Lemma 4. *If G has the isolated neighbor property for t, then the size of every monotone span program accepting $\mathcal{N}_G$ is at least $\binom{|U|}{t}$.*

Proof. We prove the lemma using Theorem 4. We take $\mathcal{C}$ to be all the pairs C_0, C_1, where $C_0 \subset U$ such that $|C_0| = t$ and $C_1 = \{v : \forall_{u \in C_0}(u, v) \notin E\}$, that is, C_1 contains all vertices that are not neighbors of any vertex in C_1. We first claim that the collection $\mathcal{C}$ satisfies the unique intersection property for $\mathcal{A}$:

- Let $(C_0, C_1) \in \mathcal{C}$ and $T = \{p_1, \ldots, p_n\} \setminus (C_0 \cup C_1)$. We need to show that $T \notin \mathcal{A}$, that is, T does not contain any minimal authorized set. Let $A \subseteq U \cap T$

be any set such that $|A| = t$. Thus, $|A| = |C_0| = t$, and there is a vertex $v \in V$ such that $v \in N(A)$ and $v \in C_1$, that is, $v \notin T$. In other words, T does not contain any minimal authorized set $A \cup N(A)$.
- Let $A \cup N(A) \in \mathcal{N}_G$ and $(C_0, C_1) \in \mathcal{C}$. First notice that $(A \cup N(A)) \cap C_0 = A \cap C_0$ and $(A \cup N(A)) \cap C_1 = N(A) \cap C_1$. Assume that $A \cap C_0 \neq \emptyset$, and let $u \in A \cap C_0$. Thus, every $v \in N(A)$ is a neighbor of u. However, every $v \in C_1$ is not a neighbor of u, and $N(A) \cap C_0 = \emptyset$.

Thus, by Theorem 4, the size of every monotone span program accepting $\mathcal{A}$ is at least $\text{rank}_{\mathbb{F}}(D)$. In this case, for every A, C_0 such that $|A| = |C_0| = t$, the entry corresponding to $A \cup N(A)$ and (C_0, C_1) is zero if $A \cap C_0 \neq \emptyset$ and is one otherwise. That is, D is the (n,t)-disjointness matrix, which has full rank over every field (see, e.g., [48, Example 2.12]).[8] The rank of D is, thus, the number of minimal authorized sets in $\mathcal{A}$, namely, $\binom{|U|}{t}$. □

As there exist graphs which satisfy the isolated neighbor property for $t = \Omega(\log n)$, e.g., the Paley Graph [1], we derive the promised lower bound.

Theorem 5. *For every n, there exists an access structure $\mathcal{N}_n$ such that every monotone span program over any field accepting it has size $n^{\Omega(\log n)}$.*

As monotone span program are equivalent to linear secret-sharing schemes [46,3], the same lower bound applies to linear secret-sharing schemes.

Corollary 1. *For every n, there exists an access structure $\mathcal{N}_n$ such that the information ratio of every linear secret-sharing scheme realizing it is $n^{\Omega(\log n)}$.*

In multi-linear secret-sharing schemes, the secret can be a vector of elements over $\mathbb{F}$, which can reduce the information ratio. However, every multi-linear secret-sharing scheme implies a linear scheme with the same share length. Thus, we obtain the following lower bound.

Corollary 2. *For every n, there exists an access structure $\mathcal{N}_n$ such that the length of the shares in every multi-linear secret-sharing scheme realizing it over the field $\mathbb{F}$ is $n^{\Omega(\log n)} \log |\mathbb{F}|$.*

Thus, already for moderate values of n, we get that the size of shares in any multi-linear secret-sharing scheme realizing $\mathcal{N}_n$ is impractical.

6 Secret-Sharing, Cryptomania, and $NP \neq coNP$

In this section we describe two unpublished results of Rudich [56], showing two surprising results connecting secret-sharing schemes to two fundamental questions in cryptography and complexity.

[8] The proof in [48] requires that $n - 2t + 1 \neq 0$ in the field $\mathbb{F}$.

6.1 Impossibility of Secret-Sharing with Efficient Sharing

In previous sections we said that a scheme is efficient if the length of the shares is polynomial in the number of parties in the access structures. This is only a necessary condition for being efficient. To use secret-sharing schemes, we should also require that the sharing process and the reconstruction are efficient. That is, when using secret-sharing schemes we want the honest parties, which share secrets and reconstruct them, to run in polynomial time.

We first consider secret-sharing where the dealer is efficient. Formally, a secret-sharing scheme $\langle \Pi, \mu \rangle$ has efficient sharing if there is an algorithm computing the mapping Π whose running time is polynomial in n (the number of parties in the access structure) and $\log |K|$ (the number of bits in the secret). Rudich [56] proved that it is unlikely that there is a secret-sharing scheme with efficient sharing realizing the Hamiltonian access structure, $\mathcal{A}_{\text{ham}}$, defined below, that is, assuming that $NP \neq coNP$, there is no such a scheme.

Definition 8. *A Hamiltonian cycle in an undirected graph $G = (V, E)$ is a simple cycle passing through all vertices in V. The Hamiltonian access structure, denoted $\mathcal{A}_{\text{ham}}$, is the access structure whose parties are edges of a complete undirected graph and its authorized sets are subsets of the edges containing a Hamiltonian cycle.*

Theorem 6 ([56]). *If $NP \neq coNP$, then there is no secret-sharing scheme with efficient reconstruction realizing $\mathcal{A}_{\text{ham}}$.*

Proof. Let $\overline{\text{HAM}} \stackrel{\text{def}}{=} \{G : G \text{ does not contain a Hamiltonian cycle}\}$. We assume in a way of contradiction that there is a secret-sharing scheme with efficient reconstruction realizing $\mathcal{A}_{\text{ham}}$ and prove that $NP = coNP$, that is, we prove that $\overline{\text{HAM}} \in NP$. The proof relies on the following simple observation: A graph $G = (V, E)$ does not contain a Hamiltonian cycle, that is, $E \notin \mathcal{A}_{\text{ham}}$, iff the shares of the parties in E do not determine the secret iff the shares of the parties in E could be generated both for the secret 0 and for the secret 1. Now, given a graph G, the witness that $G \in \overline{\text{HAM}}$ is two random strings r_0 and r_1 such that the scheme with secret $k = 0$ and random string r_0 produces the same shares for the parties in E as the scheme with secret $k = 1$ and random string r_1. □

In the above theorem, we require a very weak privacy requirement, that is, for every unauthorized set there exists shares that could be generated both for the secret 0 and the secret 1. Furthermore, we only require that the sharing is efficient and we do not care if the reconstruction is efficient. However, we require perfect correctness, that is, an authorized set can always reconstruct the correct secret.

6.2 Oblivious-Transfer Protocols from Secret-Sharing

To appreciate the result presented below we start with some background. Cryptographic protocols are built based on some assumptions. These assumption can be specific (e.g., factoring is hard) or generic (e.g., there exist one-way functions

or there exist trapdoor permutations). The minimal generic assumption is the existence of one-way functions. This assumption implies, for example, that pseudorandom generators and private-key encryption systems exist [41] and digital signatures exist [55]. However, many other tasks are not known to follow from one-way functions. Impagliazzo and Rudich [44] showed that using blackbox reductions one cannot construct oblivious-transfer protocols based on one-way functions.

The next result of Rudich [56] shows how to construct oblivious-transfer protocols based on one-way functions and an efficient secret-sharing scheme for $\mathcal{A}_{\text{ham}}$. By Theorem 6, we cannot hope for a perfect secret-sharing scheme for $\mathcal{A}_{\text{ham}}$. However, if one can construct computational secret-sharing schemes realizing $\mathcal{A}_{\text{ham}}$ based on one-way functions, then we get that one-way functions imply oblivious-transfer protocols. This will solve a major open problem in cryptography, i.e., using Impagliazzo's terminology [43], it will prove that Minicrypt = Cryptomania. As Rudich's result uses a non-blackbox reduction, such construction bypasses the impossibility result of [44].

Preliminaries. In this survey we will not define computational secret-sharing schemes. This definition can be found in [12]. In such schemes we require that the sharing and reconstruction are done in polynomial-time in the secret length and the number of parties in the access structure. Furthermore, we require that a polynomial-time adversary controlling of an unauthorized set cannot distinguish between shares of one secret and shares of another secret.

Rudich considers schemes for $\mathcal{A}_{\text{ham}}$, where the requirement on efficient reconstruction is quite weak: any authorized subset E can efficiently reconstruct the secret given that the set knows the Hamiltonian cycle in E. Thus, this weaker requirement avoids problems arising from the NP-completeness of the Hamiltonian problem.

Next, we recall the notion of 1-out-of-2 oblivious transfer [53,35]. This is a two party protocol between two parties, a sender holding two bits b_0, b_1 and a receiver holding an index $i \in \{0,1\}$. At the end of the protocol, the receiver should hold b_i without gaining any knowledge on the other bit b_{1-i}. The sender should not be able to deduce any information on i. Intuitively, the sender sends exactly one bit to the receiver, however, it is oblivious to which bit it sends. As in Section 4, we consider honest-but-curious parties. As the result of [44] already applies to this setting, constructing oblivious-transfer protocols for honest-but-curious parties is already interesting. Furthermore, by a transformation of [39], any such protocol can be transformed into a protocol secure against malicious parties assuming that one-way functions exist.

We are ready to state and prove Rudich's result.

Theorem 7 ([56]). *If one-way functions exist and an efficient secret-sharing scheme for the Hamiltonian access structure $\mathcal{A}_{\text{ham}}$ exists then oblivious-transfer protocols exist.*

Proof. Let Gen be a pseudorandom generator stretching ℓ bits to 2ℓ bits. By [41], if one-way functions exist, then such Gen exists. Define the language $L_{\text{Gen}} =$

$\{y : \exists_x \mathrm{Gen}(x) = y\}$. Clearly, $L_{\mathrm{Gen}} \in NP$. Let f be a polynomial-time reduction from L_{Gen} to Hamiltonian, that is, f can be computed in polynomial time and $y \in L_{\mathrm{Gen}}$ iff $G = f(y) \in$ Hamiltonian. Such f exists with the property that a witness to y can be efficiently translated to a witness to $G = f(x)$, that is, given $y \in L_{\mathrm{Gen}}$, a witness x for it, that is, $\mathrm{Gen}(x) = y$, and $G = f(x)$, one can find in polynomial time a Hamiltonian cycle in G. The next protocol is an oblivious-transfer protocol (for honest-but-curious parties):

Receiver's input: $i \in \{0,1\}$ and security parameter 1^ℓ.
Sender's input: b_0, b_1 and security parameter 1^ℓ.
Instructions for the receiver:
- Choose at random $x_1 \in \{0,1\}^\ell$ and compute $y_1 = \mathrm{Gen}(x_1)$.
- Choose at random $y_0 \in \{0,1\}^{2\ell}$.
- Compute $G_\sigma = f(y_\sigma)$ for $\sigma \in \{0,1\}$.
- If $i = 0$ send G_1, G_0 to the sender, else send G_0, G_1 to the sender.

Instructions for the sender:
- Let $H_0 = (V_0, E_0), H_1 = (V_1, E_1)$ be the graphs that the receiver sends.
- For $j \in \{0,1\}$, share the bit b_j using the scheme for the Hamiltonian access structure $\mathcal{A}_{\mathrm{ham}}$ for the complete graph with $|V_j|$ vertices, and send the shares of the parties corresponding to the edges in E_j to the receiver.

Instructions for the receiver: Compute a Hamiltonian cycle in G_1 from x_1 and y_1, and reconstruct b_i from the shares of this cycle for the graph $H_i = G_1$.

The privacy of the receiver is protected since the sender cannot efficiently distinguish between a string sampled according to the uniform distribution in $\{0,1\}^{2\ell}$ and an output of the generator on a string sampled uniformly in $\{0,1\}^\ell$. In particular, the sender cannot efficiently distinguish between the output of the reduction f on two such strings.

The privacy of the sender is protected against an honest-but-curious receiver since with probability at least $1 - 1/2^\ell$ the string y_0 is not in the range of Gen, thus, G_{1-i} has no Hamiltonian cycle, that is, E_i is an unauthorized set. In this case, the secret b_{1-i} cannot be efficiently computed from the shares of E_{1-i}. □

If we hope to construct an oblivious-transfer protocol using the approach of Theorem 7, then we should construct an efficient computational scheme for the Hamiltonian access structure based on the assumption that one-way functions exist. For feasibility purposes it would be interesting to construct a computational secret-sharing scheme for Hamiltonicity based on stronger cryptographic assumptions, e.g., that trapdoor permutations exist.

7 Summary and Open Problems

In this survey we consider secret-sharing schemes, a basic tool in cryptography. We show several constructions of secret-sharing schemes, starting from the scheme of [45]. We then described its generalization by [14], showing that if an

access structure can be described by a small monotone formula, then it has an efficient secret-sharing scheme. We also showed the construction of secret-sharing schemes from monotone span programs [21,46]. Monotone span programs are equivalent to linear secret-sharing schemes and are equivalent to schemes where the reconstruction is linear [3]. As every monotone formula can be transformed into a monotone span program of the same size, the monotone span program construction is a generalization of the construction of [14]. Furthermore, there are functions that have small monotone span programs and do not have small monotone formulae [2], thus, this is a strict generalization. Finally, we presented the multi-linear construction of secret-sharing schemes.

All the constructions presented in Section 3 are linear over a finite field (some of the schemes work also over finite groups, e.g., the scheme of Benaloh and Leichter). The linearity of a scheme is important in many applications, as we demonstrated in Section 4 for the construction of secure multiparty protocols for general functions. Thus, it is interesting to understand the access structures that have efficient linear secret-sharing schemes. The access structures that can efficiently be realized by linear and multi-linear secret-sharing scheme are characterized by functions that have polynomial size monotone span programs, or, more generally, multi-target monotone span programs. We would like to consider the class of access structures that can be realized by linear secret-sharing schemes with polynomial share length. As this discussion is asymptotic, we consider a sequence of access structures $\{\mathcal{A}_n\}_{n\in\mathbb{N}}$, where $\mathcal{A}_n$ has n parties. As linear algebra can be computed in NC (informally, NC is the class of problems that can be solved by parallel algorithms with polynomially many processors and poly-logarithmic running time), every sequence of access structures that has efficient linear secret-sharing schemes can be recognized by NC algorithms. For example, if $P \neq NC$, then access structures recognized by monotone P-complete problems do not have efficient linear secret-sharing schemes.

The limitations of linear secret-sharing schemes raise the question if there are non-linear secret-sharing schemes. Beimel and Ishai [6] have constructed non-linear schemes for access structures that are not known to be in P (e.g., for an access structure related to the quadratic residuosity problem over $N = pq$). Thus, non-linear schemes are probably stronger than linear schemes. Furthermore, Beimel and Ishai defined quasi-linear schemes, which are compositions of linear schemes over different fields. Beimel and Weinreb [10] showed, without any assumptions, that quasi-linear schemes are stronger than linear schemes, that is, there exists an access structure that has quasi-linear schemes with constant information ratio while every linear secret-sharing scheme realizing this access structure has super-polynomial information ratio. However, Beimel and Ishai [6] proved that if an access structure has efficient quasi-linear scheme, then it can be recognized by an NC algorithm. Thus, also the class of access structures realized by efficient quasi-linear schemes is limited.

Another non-linear construction of secret-sharing schemes is an unpublished result of Yao [69] (see also [67]). Yao showed that if an access structure can be described by a small monotone *circuit*, then it has an efficient computational

secret-sharing scheme. This generalizes the results of [14] showing that if an access structure can be described by a small monotone *formula*, then it has an efficient perfect secret-sharing scheme. We will not describe the construction of Yao in this survey.

An additional topic that we will not cover in this survey is ideal secret-sharing schemes. By Lemma 2, the size of the share of each party is at least the size of the secret. An ideal secret-sharing scheme is a scheme in which the size of the share of each party is exactly the size of the secret. For example, Shamir's scheme [58] is ideal. An access structure is ideal if it has an ideal scheme over some finite domain of secrets. For example, threshold access structures are ideal, while the access structure $\sqcap$ described in Example 5 is not ideal. Brickell [21] considered ideal schemes and constructed ideal schemes for some access structures, i.e., for hierarchical access structures. Brickell and Davenport [22] showed an interesting connection between ideal access structures and matroids, that is,

- If an access structure is ideal then it is a matroid port,
- If an access structure is a matroid port of a representable matroid, then the access structure is ideal.

Following this work, many works have constructed ideal schemes, and have studied ideal access structures and matroids. For example, Martí-Farré and Padró [49] showed that if an access structure is not a matroid port, then the information ratio of every secret-sharing scheme realizing it is at least 1.5 (compared to information ratio 1 of ideal schemes).

7.1 Open Problems

The most important open problem regarding secret-sharing schemes is settling Conjecture 1. That is,

Question 1. *Prove (or disprove) that there exists an access structure such that the information ratio of every secret-sharing scheme realizing it is* $2^{\Omega(n)}$.

A weaker version of this question is the following:

Question 2. *Prove (or disprove) that there exists an access structure such that the information ratio of every secret-sharing scheme realizing it with domain of secrets* $\{0,1\}$ *is super-polynomial in* n.

The above two questions are open even for non-explicit access structures. For linear schemes, we have super-polynomial lower bounds for explicit access structures. By counting arguments, for every n for most access structures with n parties the size of shares in every linear secret-sharing scheme realizing them is $2^{\Omega(n)}$. It is open to prove such lower bound for explicit access structures.

Question 3. *Prove that there exists an* explicit *access structure such that the information ratio of every linear secret-sharing scheme realizing it is* $2^{\Omega(n)}$.

In this survey, we describe linear and multi-linear secret-sharing schemes. It is known that multi-linear schemes are more efficient than linear schemes for small access structures, e.g., [62]. However, the possible improvement by using multi-linear schemes compared to linear schemes is open.

Question 4. *Given an access structure, what is the biggest gap between best information ratio of multi-linear schemes realizing the access structure compared to the best information ratio of linear schemes realizing the access structure?*

There are interesting access structures that we do not know if they have efficient schemes. The first access structure is the *directed connectivity* access structure whose parties are edges in a complete directed graph and whose authorized sets are sets of edges containing a path from v_1 to v_m. As there is a small monotone circuit for this access structure, by [69] it has an efficient computational scheme. It is not known if this access structure can be described by a small monotone span program and it is open if it has an efficient perfect scheme. In [9], it was proved that every monotone span program accepting the directed connectivity access structure has size $\Omega(n^{3/2})$. In comparison, the *undirected connectivity* access structure has an efficient perfect scheme [15] (see Section 3.2).

The second access structure that we do not know if it has an efficient scheme is the *perfect matching* access structure. The parties of this access structure are edges in a complete undirected graph and the authorized sets are sets of edges containing a perfect matching. It is not even known if this access structure has an efficient computational scheme as every monotone circuit for perfect matching has super-polynomial size. We remark that an efficient scheme for this access structure implies an efficient scheme for the directed connectivity access structure.

The third interesting family of access structures is *weighted threshold* access structures. In such an access structure each party has a weight and there is some threshold. A set of parties is authorized if the sum of the weights of the parties in the set is bigger than the threshold. For these access structures there is an efficient computational scheme [11] and a perfect scheme with $n^{O(\log n)}$ long shares. It is open if these access structures have a perfect scheme with polynomial shares. Furthermore, it is open if they can be described by polynomial size monotone formulae.

Acknowledgment

I would like to thank Benny Chor for introducing me to the field of secret-sharing schemes and guiding me in the early stages of my career. I would also like to thank my co-authors in papers related to secret sharing: Mike Burmester, Yvo Desmedt, Matt Franklin, Anna Gál, Yuval Ishai, Eyal Kushilevitz, Noam Livne, Ilan Orlov, Carles Padró, Anat Paskin, Mike Paterson, Tamir Tassa, and Enav Weinreb. I learned a lot from working with them. Finally, thanks to Moni Naor for telling me about the unpublished results of Rudich presented in Section 6.

References

1. Alon, N., Goldreich, O., Håstad, J., Peralta, R.: Simple constructions of almost k-wise independent random variables. Random Structures & Algorithms 3, 289–304 (1992)
2. Babai, L., Gál, A., Wigderson, A.: Superpolynomial lower bounds for monotone span programs. Combinatorica 19(3), 301–319 (1999)
3. Beimel, A.: Secure Schemes for Secret Sharing and Key Distribution. PhD thesis, Technion (1996), `http://www.cs.bgu.ac.il/~beimel/pub.html`
4. Beimel, A., Chor, B.: Universally ideal secret sharing schemes. IEEE Trans. on Information Theory 40(3), 786–794 (1994)
5. Beimel, A., Gál, A., Paterson, M.: Lower bounds for monotone span programs. Computational Complexity 6(1), 29–45 (1997); Conference version: FOCS 1995
6. Beimel, A., Ishai, Y.: On the power of nonlinear secret-sharing. SIAM J. on Discrete Mathematics 19(1), 258–280 (2005)
7. Beimel, A., Livne, N., Padró, C.: Matroids can be far from ideal secret sharing. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 194–212. Springer, Heidelberg (2008)
8. Beimel, A., Orlov, I.: Secret sharing and non-shannon information inequalities. IEEE Trans. on Information Theory (2011); Preliminary version Reingold, O. (ed.): TCC 2009. LNCS, vol. 5444, pp. 539–557. Springer, Heidelberg (2009)
9. Beimel, A., Paskin, A.: On linear secret sharing for connectivity in directed graphs. In: Ostrovsky, R., De Prisco, R., Visconti, I. (eds.) SCN 2008. LNCS, vol. 5229, pp. 172–184. Springer, Heidelberg (2008)
10. Beimel, A., Weinreb, E.: Separating the power of monotone span programs over different fields. SIAM J. on Computing 34(5), 1196–1215 (2005)
11. Beimel, A., Weinreb, E.: Monotone circuits for monotone weighted threshold functions. Inform. Process. Lett. 97(1), 12–18 (2006); Conference version: Proc. of 20th Annu. IEEE Conf. on Computational Complexity, pp. 67–75 (2005)
12. Bellare, M., Rogaway, P.: Robust computational secret sharing and a unified account of classical secret-sharing goals. In: Proc. of the 14th ACM Conference on Computer and Communications Security, pp. 172–184 (2007)
13. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for noncryptographic fault-tolerant distributed computations. In: Proc. of the 20th ACM Symp. on the Theory of Computing, pp. 1–10 (1988)
14. Benaloh, J.C., Leichter, J.: Generalized secret sharing and monotone functions. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 27–35. Springer, Heidelberg (1990)
15. Benaloh, J.C., Rudich, S.: Private communication (1989)
16. Bertilsson, M., Ingemarsson, I.: A construction of practical secret sharing schemes using linear block codes. In: Zheng, Y., Seberry, J. (eds.) AUSCRYPT 1992. LNCS, vol. 718, pp. 67–79. Springer, Heidelberg (1993)
17. Blakley, G.R.: Safeguarding cryptographic keys. In: Merwin, R.E., Zanca, J.T., Smith, M. (eds.) Proc. of the 1979 AFIPS National Computer Conference. AFIPS Conference Proceedings, vol. 48, pp. 313–317. AFIPS Press (1979)
18. Blundo, C., De Santis, A., De Simone, R., Vaccaro, U.: Tight bounds on the information rate of secret sharing schemes. Designs, Codes and Cryptography 11(2), 107–122 (1997)
19. Blundo, C., De Santis, A., Gargano, L., Vaccaro, U.: On the information rate of secret sharing schemes. Theoretical Computer Science 154(2), 283–306 (1996)

20. Blundo, C., De Santis, A., Vaccaro, U.: On secret sharing schemes. Inform. Process. Lett. 65(1), 25–32 (1998)
21. Brickell, E.F.: Some ideal secret sharing schemes. Journal of Combin. Math. and Combin. Comput. 6, 105–113 (1989)
22. Brickell, E.F., Davenport, D.M.: On the classification of ideal secret sharing schemes. J. of Cryptology 4(73), 123–134 (1991)
23. Capocelli, R.M., De Santis, A., Gargano, L., Vaccaro, U.: On the size of shares for secret sharing schemes. J. of Cryptology 6(3), 157–168 (1993)
24. Chaum, D., Crépeau, C., Damgård, I.: Multiparty unconditionally secure protocols. In: Proc. of the 20th ACM Symp. on the Theory of Computing, pp. 11–19 (1988)
25. Chor, B., Goldwasser, S., Micali, S., Awerbuch, B.: Verifiable secret sharing and achieving simultaneity in the presence of faults (extended abstract). In: Proc. of the 26th IEEE Symp. on Foundations of Computer Science, pp. 383–395 (1985)
26. Chor, B., Kushilevitz, E.: Secret sharing over infinite domains. J. of Cryptology 6(2), 87–96 (1993)
27. Cover, T.M., Thomas, J.A.: Elements of Information Theory. John Wiley & Sons, Chichester (1991)
28. Cramer, R., Damgård, I.B., Maurer, U.M.: General secure multi-party computation from any linear secret-sharing scheme. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 316–334. Springer, Heidelberg (2000)
29. Csirmaz, L.: The size of a share must be large. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 223–231. Springer, Heidelberg (1995); Journal version in: J. of Cryptology 10(4), 223–231 (1997)
30. Csirmaz, L.: The dealer's random bits in perfect secret sharing schemes. Studia Sci. Math. Hungar. 32(3-4), 429–437 (1996)
31. Desmedt, Y.G., Frankel, Y.: Shared generation of authenticators and signatures. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 457–469. Springer, Heidelberg (1992)
32. van Dijk, M.: A linear construction of perfect secret sharing schemes. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 23–34. Springer, Heidelberg (1995)
33. van Dijk, M.: On the information rate of perfect secret sharing schemes. Designs, Codes and Cryptography 6, 143–169 (1995)
34. van Dijk, M., Kevenaar, T., Schrijen, G.-J., Tuyls, P.: Improved constructions of secret sharing schemes by applying $(\lambda], \omega)$-decompositions. Inform. Process. Lett. 99(4), 154–157 (2006)
35. Even, S., Goldreich, O., Lempel, A.: A randomized protocol for signing contracts. CACM 28(6), 637–647 (1985)
36. Gál, A.: A characterization of span program size and improved lower bounds for monotone span programs. Computational Complexity 10(4), 277–296 (2002)
37. Gál, A., Pudlák, P.: Monotone complexity and the rank of matrices. Inform. Process. Lett. 87, 321–326 (2003)
38. Gennaro, R., Rabin, M.O., Rabin, T.: Simplified vss and fact-track multiparty computations with applications to threshold cryptography. In: Proc. of the 17th ACM Symp. on Principles of Distributed Computing, pp. 101–111 (1998)
39. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game. In: Proc. of the 19th ACM Symp. on the Theory of Computing, pp. 218–229 (1987)
40. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proc. of the 13th ACM Conference on Computer and Communications Security, pp. 89–98 (2006)

41. Håstad, J., Impagliazzo, R., Levin, L.A., Luby, M.: Construction of a pseudo-random generator from any one-way function. SIAM J. on Computing 28(4), 1364–1396 (1999)
42. Hirt, M., Maurer, U.: Player simulation and general adversary structures in perfect multiparty computation. J. of Cryptology 13(1), 31–60 (2000)
43. Impagliazzo, R.: A personal view of average-case complexity. In: Proc. of the 10th IEEE Structure in Complexity Theory, pp. 134–147 (1995)
44. Impagliazzo, R., Rudich, S.: Limits on the provable consequences of one-way permutations. In: Proc. of the 21st ACM Symp. on the Theory of Computing, pp. 44–61 (1989)
45. Ito, M., Saito, A., Nishizeki, T.: Secret sharing schemes realizing general access structure. In: Proc. of the IEEE Global Telecommunication Conf., Globecom 1987, pp. 99–102 (1987); Journal version: Multiple assignment scheme for sharing secret. J. of Cryptology, 6(1), 15–20 (1993)
46. Karchmer, M., Wigderson, A.: On span programs. In: Proc. of the 8th IEEE Structure in Complexity Theory, pp. 102–111 (1993)
47. Karnin, E.D., Greene, J.W., Hellman, M.E.: On secret sharing systems. IEEE Trans. on Information Theory 29(1), 35–41 (1983)
48. Kushilevitz, E., Nisan, N.: Communication Complexity. Cambridge University Press, Cambridge (1997)
49. Martí-Farré, J., Padró, C.: On secret sharing schemes, matroids and polymatroids. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 273–290. Springer, Heidelberg (2007)
50. Matúš, F.: Infinitely many information inequalities. In: IEEE International Symposium on Information Theory 2007, pp. 41–44 (2007)
51. Metcalf-Burton, J.R.: Improved upper bounds for the information rates of the secret sharing schemes induced by the Vamos matroid. Technical Report abs/0809.3010, CoRR (2008)
52. Naor, M., Wool, A.: Access control and signatures via quorum secret sharing. IEEE Transactions on Parallel and Distributed Systems 9(1), 909–922 (1998)
53. Rabin, M.O.: How to exchange secrets by oblivious transfer. Technical Report TR-81, Harvard Aiken Computation Laboratory (1981), Available online in the Cryptology ePrint Archive, Report 2005/187, `http://eprint.iacr.org/2005/187`
54. Rabin, M.O.: Randomized Byzantine generals. In: Proc. of the 24th IEEE Symp. on Foundations of Computer Science, pp. 403–409 (1983)
55. Rompel, J.: One-way functions are necessary and sufficient for secure signatures. In: Proc. of the 22nd ACM Symp. on the Theory of Computing, pp. 387–394 (1990)
56. Rudich, S.: Private communication, via M. Naor (1989)
57. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
58. Shamir, A.: How to share a secret. Communications of the ACM 22, 612–613 (1979)
59. Shankar, B., Srinathan, K., Rangan, C.P.: Alternative protocols for generalized oblivious transfer. In: Rao, S., Chatterjee, M., Jayanti, P., Murthy, C.S.R., Saha, S.K. (eds.) ICDCN 2008. LNCS, vol. 4904, pp. 304–309. Springer, Heidelberg (2008)
60. Simmons, G.J.: How to (really) share a secret. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 390–448. Springer, Heidelberg (1990)
61. Simmons, G.J., Jackson, W., Martin, K.M.: The geometry of shared secret schemes. Bulletin of the ICA 1, 71–88 (1991)
62. Simonis, J., Ashikhmin, A.: Almost affine codes. Designs, Codes and Cryptography 14(2), 179–197 (1998)

63. Stinson, D.R.: Decomposition construction for secret sharing schemes. IEEE Trans. on Information Theory 40(1), 118–125 (1994)
64. Tassa, T.: Hierarchical threshold secret sharing. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 473–490. Springer, Heidelberg (2004)
65. Tassa, T.: Generalized oblivious transfer by secret sharing. Designs, Codes and Cryptography 58 (2011)
66. Tassa, T., Dyn, N.: Multipartite secret sharing by bivariate interpolation. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 288–299. Springer, Heidelberg (2006)
67. Vinod, V., Narayanan, A., Srinathan, K., Pandu Rangan, C., Kim, K.: On the power of computational secret sharing. In: Johansson, T., Maitra, S. (eds.) INDOCRYPT 2003. LNCS, vol. 2904, pp. 162–176. Springer, Heidelberg (2003)
68. Waters, B.: Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. Technical Report 2008/290, Cryptology ePrint Archive (2008), `http://eprint.iacr.org/`
69. Yao, A.C.: Unpublished manuscript. Presented at Oberwolfach and DIMACS Workshops (1989)
70. Yeung, R.W.: Information Theory and Network Coding. Springer, Heidelberg (2008)
71. Zhang, Z., Yeung, R.W.: On characterization of entropy function via information inequalities. IEEE Trans. on Information Theory 44(4), 1440–1452 (1998)

Lattice Codes for the Gaussian Wiretap Channel

Jean-Claude Belfiore[1], Frédérique Oggier[2], and Patrick Solé[1]

[1] Dept. of Communications & Electronics
Telecom ParisTech, CNRS/LTCI
Paris, France
[2] Division of Mathematical Sciences School of Physical and Mathematical Sciences
Nanyang Technological University
Singapore

Abstract. It has been shown recently that coding for the Gaussian Wiretap Channel can be done with nested lattices. A fine lattice intended to the legitimate user must be designed as a usual lattice code for the Gaussian Channel, while a coarse lattice is added to introduce confusion at the eavesdropper, whose theta series must be minimized. We study, here, the behavior of this invariant for a class of lattices.

1 Introduction

The wiretap channel was introduced by Wyner [1] as a discrete memoryless broadcast channel where the sender, Alice, transmits confidential messages to a legal receiver Bob, in the presence of an eavesdropper Eve. Wyner defined the perfect secrecy capacity as the maximum amount of information that Alice can send to Bob while insuring that Eve gets a negligeable amount of information. He also described a generic coding strategy known as coset coding. While coset coding has been used in many coding scenarios (for ex. [2,3]), Wyner used it to encode both data and random bits to confuse the eavesdropper. The question of determining the secrecy capacity of many classes of channels has been addressed extensively recently, yielding a plethora of information theoretical results on secrecy capacity.

There is a sharp contrast with the situation of wiretap code designs, where very little is known. The most exploited approach to get practical codes so far has been to use LDPC codes (for example [4] for binary erasure and symmetric channels, [5] for Gaussian channels with binary inputs). We also note that wiretap II codes have been extended to more general settings such as network coding in [6]. Finally, lattice codes for Gaussian channels have been considered from an information theoretical point of view in [7].

In [8], a design criterion for constructing explicit lattice codes, has been proposed, based on the analysis of Eve's correct decision probability. This design criterion relies on a new lattice invariant called "secrecy gain" based on theta series. In this paper, we analyze the secrecy gain of unimodular lattices.

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 47–61, 2011.
© Springer-Verlag Berlin Heidelberg 2011

2 Notations and Previous Results

2.1 Notations and System Model

We analyze more deeply the secrecy gain introduced in [8] for even unimodular lattices and give the asymptotic behavior of this secrecy gain when the dimension of the lattices grows to infinity. Figure 1 gives the model considered in this

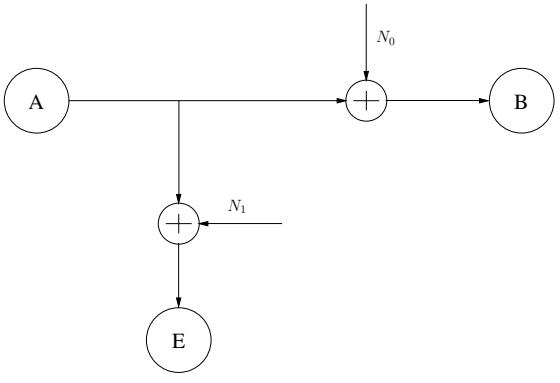


Fig. 1. The Gaussian Wiretap Channel

paper where Alice wants to send data to Bob on a Gaussian channel whose noise variance is given by σ_b^2. Eve is the eavesdropper trying to intercept data through another Gaussian channel whose noise variance is σ_e^2. In order to have a positive secrecy capacity, we will assume that $\sigma_e^2 > \sigma_b^2$. Bits are transmitted by Alice at a rate equal to $R = R_s + R_r$ where R_s is the secrecy rate of this transmission and R_r is the rate of pseudo-random bits. Indeed, we use Wyner's generic coding strategy [9]. We give the remaining parameters,

- Λ_b is the fine lattice (used to minimize Bob's probability of error)
- Λ_e is the coarse lattice (used to minimize Eve's probability of correct decision)
- n is the dimension of both lattices
- $\mathcal{V}(\Lambda_b)$ (resp. $\mathcal{V}(\Lambda_e)$) is the fundamental parallelotope of Λ_b (resp. Λ_e)
- $\mathrm{Vol}(\mathcal{P})$ is the volume of $\mathcal{P}$

Data bits label cosets in Λ_b/Λ_e while pseudo-random bits label points of Λ_e. The reader can refer to [8] for a more detailed description of the coding scheme. Still according to [8], and under the assumption of a moderate to high secrecy rate, the expression of the probability of correct decision at the eavesdropper can be expressed as

$$P_{c,e} \simeq \left(\frac{1}{\sqrt{2\pi}\sigma_e}\right)^n \mathrm{Vol}\left(\mathcal{V}(\Lambda_b)\right) \sum_{\mathbf{r}\in\Lambda_e} e^{-\frac{\|\mathbf{r}\|^2}{2\sigma_e^2}}. \quad (1)$$

In eq. (1), we recognize the theta series of lattice Λ_e.

2.2 Theta Series of a Lattice

Definition 1. *Let Λ be a Euclidean lattice, then the theta series of Λ is [10]*

$$\Theta_\Lambda(z) \triangleq \sum_{\boldsymbol{x}\in\Lambda} q^{\|\boldsymbol{x}\|^2}, q = e^{i\pi z}, \mathrm{Im}(z) > 0 \tag{2}$$

Some exceptional lattices have theta series that can be expressed as functions of the Jacobi theta functions $\vartheta_i(q)$, $i = 2, 3, 4$ with

$$\vartheta_2(q) = \sum_{n=-\infty}^{+\infty} q^{\left(n+\frac{1}{2}\right)^2}$$

$$\vartheta_3(q) = \sum_{n=-\infty}^{+\infty} q^{n^2}$$

$$\vartheta_4(q) = \sum_{n=-\infty}^{+\infty} (-1)^n q^{n^2}$$

For instance, table 1 gives the theta series of some exceptional lattices.

Table 1. Theta series of some lattices

Lattice Λ	Theta series Θ_Λ
Cubic lattice $\mathbb{Z}^n$	ϑ_3^n
D_n	$\frac{1}{2}\left(\vartheta_3^n + \vartheta_4^n\right)$
Gosset lattice E_8	$\frac{1}{2}\left(\vartheta_2^8 + \vartheta_3^8 + \vartheta_4^8\right)$

2.3 Minimization of the Theta Series

One problem that arises naturally when studying theta series is the following. In eq. (1), set $y = iz$ and restrict to real values of y. We are now interested in studying

$$\Theta_\Lambda(y) = \sum_{\boldsymbol{x}\in\Lambda} q^{\|\boldsymbol{x}\|^2}, q = e^{-\pi y}, y > 0.$$

Equation (1), giving Eve's probability of correct decision, can be written as

$$P_{c,e} \simeq \left(\frac{1}{\sqrt{2\pi}\sigma_e}\right)^n \mathrm{Vol}\left(\mathcal{V}\left(\Lambda_b\right)\right) \Theta_{\Lambda_e}\left(\frac{1}{2\pi\sigma_e^2}\right) \tag{3}$$

So, for a given dimension n, the problem to solve is to find a lattice Λ^{opt} that minimizes $\Theta_\Lambda(y)$ for a given value of y in order to minimize expression (3).

3 The Secrecy Gain

3.1 Definitions

We recall here some definitions given in [8].

We remark that, if we do not use any specific coarse lattice Λ_e, we can assume that Λ_e is equal to a scaled version of $\mathbb{Z}^n$ with same volume as Λ_e. Consequently, for a lattice Λ, it is natural to define the secrecy function. For a lattice Λ with fundamental volume $\mathrm{Vol}\left(\mathcal{V}\left(\Lambda\right)\right)$, we have

Definition 2. *Let Λ be an $n-$dimensional lattice. The secrecy function of Λ is*

$$\Xi_\Lambda(y) \triangleq \frac{\Theta_{\lambda\mathbb{Z}^n}(y)}{\Theta_\Lambda(y)} = \frac{\vartheta_3(\lambda^2 y)^n}{\Theta_\Lambda(y)}$$

where $\lambda = \mathrm{Vol}\left(\mathcal{V}\left(\Lambda_b\right)\right)^{\frac{1}{n}}$ and defined for $y > 0$.

Then of course, as we want to minimize the expression of Eve's probability of correct decision in eq. (3), we are interested in the maximum value of the secrecy function. So, we define the secrecy gain,

Definition 3. *The secrecy gain of an $n-$dimensional lattice Λ is*

$$\chi_\Lambda \triangleq \sup_{y>0} \Xi_\Lambda(y).$$

For lattices equivalent to their dual, the secrecy function exhibits a multiplicative symmetry point at $y_0 = \mathrm{Vol}\left(\mathcal{V}\left(\Lambda\right)\right)^{-\frac{2}{n}}$ for which we conjecture that

$$\Xi_\Lambda\left(y_0\right) = \chi_\Lambda.$$

3.2 The Secrecy Gain of Unimodular Lattices

Theta series are difficult to analyze. Nevertheless, for some lattices, these functions have nice properties. It is the case of even unimodular lattices whose theta series are modular forms with integer weight. We mainly restrict this paper to the study of even unimodular lattices and will use tools from modular forms.

Definitions and formulas. We recall the definition of an integral lattice [10],

Definition 4. *A lattice Λ is integral if its Gram matrix has entries in $\mathbb{Z}$. Note that an integral lattice has the property*

$$\Lambda \subseteq \Lambda^\star \subseteq \frac{1}{Vol\left(\mathcal{V}\left(\Lambda\right)\right)^2}\Lambda$$

From this definition, we can now define unimodular lattices,

Definition 5. *A lattice Λ is unimodular if*

1. *Λ is integral*
2. *Λ is equal to its dual*

Note that a unimodular lattice has fundamental volume equal to 1.

Let $\Lambda^\star$ be the dual lattice of the $n-$dimensional lattice Λ. Then Jacobi's formula [10] gives the theta series of $\Lambda^\star$ as a function of the theta series of Λ,

$$\Theta_{\Lambda^\star}(y) = \mathrm{Vol}\left(\mathcal{V}\left(\Lambda\right)\right) y^{-\frac{n}{2}} \Theta_\Lambda\left(\frac{1}{y}\right) \tag{4}$$

If Λ is unimodular, then using (4), we deduce

$$\Theta_\Lambda(y) = \Theta_{\Lambda^\star}(y) = y^{-\frac{n}{2}} \Theta_\Lambda\left(\frac{1}{y}\right).$$

So, since $\mathbb{Z}^n$ itself is unimodular, the secrecy function of Λ has the property,

$$\Xi_\Lambda(y) = \Xi_\Lambda\left(\frac{1}{y}\right).$$

If we express y in decibel (in our case, $y = \frac{1}{2\pi\sigma_e^2}$ and is related to Eve's signal to noise ratio), then the secrecy function becomes an even function.

Conjecture 1. The secrecy gain of unimodular lattices is achieved by the secrecy function at $y = 1$.

Using conjecture 1 in what follows, we can evaluate the secrecy gain of unimodular lattices as

$$\chi_\Lambda = \Xi_\Lambda(1)$$

Some formulas. Some formulas are useful to calculate the secrecy gain of unimodular lattices. The most important ones, found in [11], are

$$\begin{aligned} \vartheta_2\left(e^{-\pi}\right) &= \vartheta_4\left(e^{-\pi}\right) \\ \vartheta_3\left(e^{-\pi}\right) &= \sqrt[4]{2}\vartheta_4\left(e^{-\pi}\right) \end{aligned} \tag{5}$$

Secrecy gain of some exceptional unimodular lattices

Gosset Lattice E_8. E_8 is unimodular even. From table 1 and eq. (5), we get

$$\begin{aligned} \frac{1}{\Xi_{E_8}(1)} &= \frac{\frac{1}{2}\left(\vartheta_2(e^{-\pi})^8 + \vartheta_3(e^{-\pi})^8 + \vartheta_4(e^{-\pi})^8\right)}{\vartheta_3(e^{-\pi})^8} \\ &= \tfrac{1}{2}\left(1 + \frac{1}{4} + \frac{1}{4}\right) \\ &= \frac{3}{4} \end{aligned}$$

We deduce, then, the secrecy gain of E_8,

$$\boxed{\chi_{E_8} = \Xi_{E_8}(1) = \frac{4}{3} = 1.33333}$$

As an illustration, figure 2 gives the secrecy function of E_8.

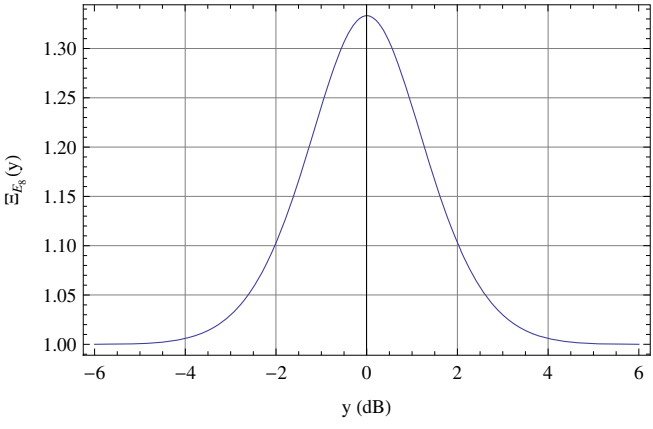


Fig. 2. Secrecy function of E_8

Leech Lattice Λ_{24}. Λ_{24} is also unimodular even. From table 1, we get (with simplified notations)

$$\begin{aligned}\frac{1}{\Xi_{\Lambda_{24}}(1)} &= \frac{\frac{1}{8}\left(\vartheta_2^8+\vartheta_3^8+\vartheta_4^8\right)^3 - \frac{45}{16}\vartheta_2^8\vartheta_3^8\vartheta_4^8}{\vartheta_3^{24}} \\ &= \frac{27}{2^6} - \frac{45}{2^8} \\ &= \frac{63}{256}\end{aligned}$$

We deduce, then, the secrecy gain of Λ_{24},

$$\boxed{\chi_{\Lambda_{24}} = \Xi_{\Lambda_{24}}(1) = \frac{256}{63} = 4.0635}$$

As an illustration, figure 3gives the secrecy function of Λ_{24}.

3.3 Higher Dimension Unimodular Extremal Lattices

E_8 and Λ_{24} are extremal even unimodular lattices in dimensions 8 and 24 respectively [10]. Extremal means that their minimum distance is maximal for a

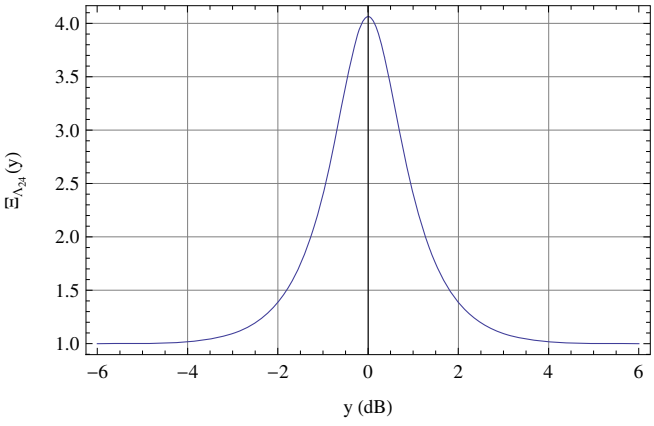


Fig. 3. Secrecy function of Λ_{24}

Table 2. Theta series of extremal lattices

Dimension	Lattice Λ	Θ_Λ
8	E_8	E_4
24	Λ_{24}	$E_4^3 - 720\Delta$
32	BW_{32}	$E_4^4 - 960E_4\Delta$
48	P_{48}	$E_4^6 - 1440E_4^3\Delta + 125280\Delta^2$
72	L_{72}	$E_4^9 - 2160E_4^6\Delta + 965520E_4^3\Delta^2 - 27302400\Delta^3$
80	L_{80}	$E_4^{10} - 2400E_4^7\Delta + 1360800E_4^4\Delta^2 - 103488000E_4\Delta^3$

given dimension [10]. We can give same type of results for extremal even unimodular lattices of higher dimensions. For instance, we can derive the secrecy functions and secrecy gains of extremal even unimodular lattices in dimensions 32, 48, and 72 using derivations of [12]. The same can be done in dimension 80 by solving a linear system [13]. Please note that, until now, nobody knows if an extremal lattice in dimension 72 exists. Results are summarized in table 2. Here we introduce the function

$$\Delta(q) = \frac{E_4^3(q) - E_6^2(q)}{12^3}$$

where E_k are the Eisenstein series [13] defined as

$$\begin{aligned} E_k(q) &= 1 + \frac{2}{\zeta(1-k)} \sum_{m=1}^{+\infty} m^{k-1} \frac{q^m}{1-q^m} \\ &= 1 - \frac{2k}{B_k} \sum_{m=1}^{+\infty} m^{k-1} \frac{q^m}{1-q^m} \end{aligned} \tag{6}$$

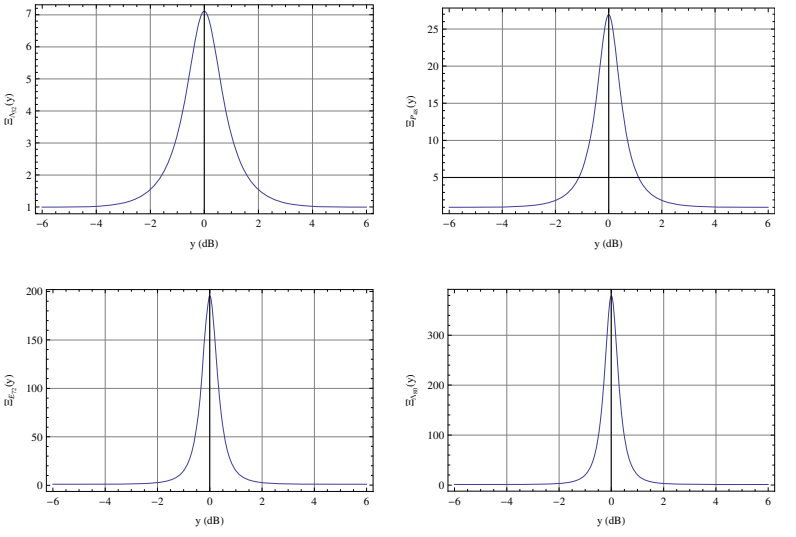


Fig. 4. Secrecy functions of extremal lattices in dimensions 32, 48, 72 and 80

where B_k are the Bernouilli numbers [14] and $\zeta(s)$ is the Riemann zeta function

$$\zeta(s) = \sum_{k=1}^{\infty} \frac{1}{k^s}.$$

Relations with Jacobi functions are (in symbolic notation)

$$\begin{cases} E_4 &= \frac{1}{2}\left(\vartheta_2^8 + \vartheta_3^8 + \vartheta_4^8\right) \\ \Delta &= \frac{1}{256}\vartheta_2^8\vartheta_3^8\vartheta_4^8 \end{cases}$$

and give rise to the expressions of theta series evaluated below.

Barnes Wall lattice BW_{32}. In dimension 32, Barnes-Wall lattice BW_{32} is an extremal lattice. We have

$$\begin{aligned} \Theta_{BW_{32}} = \frac{1}{16}\left(\vartheta_2^8 + \vartheta_3^8 + \vartheta_4^8\right)\Big[&\left(\vartheta_2^8 + \vartheta_3^8 + \vartheta_4^8\right)^3 \\ &-30 \cdot \vartheta_2^8 \cdot \vartheta_3^8 \cdot \vartheta_4^8\Big] \end{aligned}$$

so,

$$\begin{aligned} \frac{1}{\Xi_{BW_{32}}(1)} &= \frac{1}{16}\left(1+\frac{1}{2}\right)\left[\left(1+\frac{1}{2}\right)^3 - 30 \cdot \frac{1}{16}\right] \\ &= \frac{9}{64}. \end{aligned}$$

Hence,

$$\chi_{BW_{32}} = \frac{64}{9} \simeq 7.11$$

Extremal lattices in dimensions 48, 72 and 80. In the same way, from table 2, we can compute the secrecy gain for extremal even unimodular lattices in dimensions 48, 72 and 80. We have

$$\begin{aligned}\chi_{\Lambda_{48}} &= \tfrac{524288}{19467} \simeq 26.93\\ \chi_{\Lambda_{72}} &= \tfrac{134217728}{685881} \simeq 195.69\\ \chi_{\Lambda_{80}} &= \tfrac{536870912}{1414413} \simeq 379.57\end{aligned}$$

Table 3 summarizes all these results.

Table 3. Secrecy gains of extremal lattices

Dimension	8	24	32	48	72	80
Secrecy gain	1.3	4.1	7.11	26.9	195.7	380

4 Asymptotic Analysis

We propose, here to find a lower bound of the best secrecy gain as a function of the dimension n, and deduce some asymptotic results (when n is large enough). For a fixed dimension n, we compute bounds on the theta series of an optimal unimodular lattice. By optimal, we mean a lattice which maximizes the secrecy gain. We will use the Siegel-Weil formula to compute these bounds.

4.1 A Siegel-Weil Formula for Theta Series of Even Unimodular Lattices

Let $n \equiv 0 \pmod 8$, Ω_n be the set of all inequivalent even unimodular $n-$dimensional lattices. Let $k = n/2$. Then, one has [14]

$$\sum_{\Lambda\in\Omega_n} \frac{\Theta_\Lambda(q)}{|\mathrm{Aut}(\Lambda)|} = M_n \cdot E_k(q)$$

where

$$M_n = \sum_{\Lambda\in\Omega_n} \frac{1}{|\mathrm{Aut}(\Lambda)|}$$

and $E_k(q)$ is the Eisenstein series with weight k even whose expression is given in eq. (6).

Let $\Theta_{\min}^{(n)} = \min_{\Lambda\in\Omega_n} \Theta_\Lambda$. Then

$$\Theta_{\min}^{(n)} M_n \leq \sum_{\Lambda\in\Omega_n} \frac{\Theta_\Lambda}{|\mathrm{Aut}(\Lambda)|} = M_n E_k$$

giving rise to

$$\Theta_{\min}^{(n)} \leq E_k.$$

Define

$$\chi_n \triangleq \max_{\Lambda \in \Omega_n} \chi_\Lambda = \frac{\vartheta_3^n\left(e^{-\pi}\right)}{\Theta_{\min}^{(n)}\left(e^{-\pi}\right)}$$

then we get,

$$\boxed{\chi_n \geq \frac{\vartheta_3^n\left(e^{-\pi}\right)}{E_k\left(e^{-2\pi}\right)}}$$

4.2 Limit of E_k

Assume q to be a real number $0 < q < 1$. We have

$$E_k(q) = 1 + \frac{2k}{|B_k|} \sum_{m=1}^{+\infty} m^{k-1} \frac{q^m}{1-q^m}$$

Replacing q by $e^{-2\pi}$ gives

$$E_k\left(e^{-2\pi}\right) = 1 + \frac{2k}{|B_k|} \sum_{m=1}^{+\infty} \frac{m^{k-1}}{e^{2\pi m} - 1}$$

which converges to 2 when k is a multiple of 4 that tends to infinity [15]. Moreover, according to [11], we have

$$\vartheta_3\left(e^{-\pi}\right) = \frac{\pi^{\frac{1}{4}}}{\Gamma\left(\frac{3}{4}\right)} \simeq 1.086 > 1$$

so,

$$\boxed{\chi_n \gtrsim \frac{1}{2}\left(\frac{\pi^{\frac{1}{4}}}{\Gamma\left(\frac{3}{4}\right)}\right)^n \simeq \frac{1.086^n}{2}} \tag{7}$$

which tends exponentially to infinity. Figure 5 gives the asymptotic expression of the secrecy gain as a function of the dimension n, as well as points corresponding to extremal lattices in dimensions $8, 16, 24, 32, 48, 72$ and 80.

4.3 Consequences

We proved that there exists a family of even unimodular lattices whose secrecy gains exponentially grows up with the dimension, which means that Eve's probability of correct decision exponentially tends to 0. But as we can remark in figure 4, around its maximum, the secrecy function becomes sharper and sharper when n grows up, which means that, for high dimensions, the communication system absolutely has to operate at $y = 1$. We show now, in section 5, that it is possible to do the same with any integral lattice.

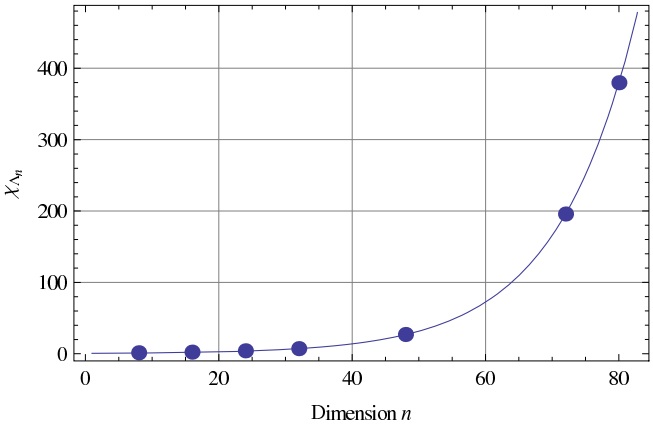


Fig. 5. Lower bound of the minimal secrecy gain as a function of n from Siegel-Weil formula. Points correspond to extremal lattices.

5 Integral Lattices

5.1 Level of a Lattice (*resp.* a Quadratic Form)

There is an equivalence between lattices and quadratic forms. In what follows, we will either deal with lattices or with quadratic forms, thanks to this equivalence. Some classical definitions follow. Let

$$f(\boldsymbol{x}) = \sum_{i,j} f_{ij} x_i x_j \tag{8}$$

be a quadratic form. Then, f is integral whenever

$$f_{ij} \in \mathbb{Z}, \forall i, j.$$

The form is primitive when

$$\gcd(f_{i,j}) = 1$$

Now, let $f(\boldsymbol{x})$ be a primitive integral positive definite quadratic form in an even number

$$n = 2k \geq 4$$

of variables. We write f in the shape

$$f(\boldsymbol{x}) = \frac{1}{2} \boldsymbol{x}^t \cdot \boldsymbol{F} \cdot \boldsymbol{x}$$

so that the elements of matrix $\boldsymbol{F}$ are integers and those on the principal diagonal are even. We define the *level* of f as being the integer $N > 0$ such that

$$N\boldsymbol{F}^{-1}$$

corresponds in the same way to a primitive integer-valued form. It may be easily verified that N and $\det \boldsymbol{F}$ have the same prime factors.

5.2 Dirichlet Character

We define now the character

$$\chi(a) = \left(\frac{(-1)^k \det \boldsymbol{F}}{a} \right) \tag{9}$$

where the symbol on the right side is the Kronecker symbol and a is any integer. The definition of the Kronecker symbol $\left(\frac{b}{n}\right)$ follows [16, Chap. 4]. If $n = p$ is an odd prime, then, for any integer b,

$$\left(\frac{b}{p}\right) = \begin{cases} 0 & \text{if } \gcd(b,p) \neq 1, \\ 1 & \text{if } b \text{ is a square mod } p, \\ -1 & \text{if } b \text{ is not a square mod } p. \end{cases}$$

If $p = 2$, then

$$\left(\frac{b}{2}\right) = \begin{cases} 0 & \text{if } b \text{ is even}, \\ 1 & \text{if } b \equiv \pm 1 \pmod 8, \\ -1 & \text{if } b \equiv \pm 3 \pmod 8. \end{cases}$$

More generally, if $n = \prod_i p_i^{e_i}$ where the p_i are prime, then

$$\left(\frac{b}{n}\right) = \prod_i \left(\frac{b}{p_i}\right)^{e_i}.$$

It can be shown that the character χ, defined in eq. (9), is a Dirichlet character of modulus N [17, Appendix B].

5.3 Theta Series as a Modular Form

Consider the quadratic form of eq. (8) with n variable ($n = 2k \geq 4$). This quadratic form is associated to a $n-$dimensional Euclidean lattice Λ via its Gram matrix. Define the theta series of Λ as

$$\Theta_\Lambda(q) = \sum_{\boldsymbol{x} \in \Lambda} q^{\|x\|^2} = \sum_{\boldsymbol{u} \in \mathbb{Z}^n} q^{f(\boldsymbol{u})}.$$

Λ is said to be integral, of level N, as the quadratic form f.

Theorem 1. *The theta series of Λ is a modular form, for the congruence group*

$$\Gamma_0(N) = \left\{ \begin{bmatrix} a & b \\ c & d \end{bmatrix} \in \Gamma : \begin{bmatrix} a & b \\ c & d \end{bmatrix} \equiv \begin{bmatrix} \star & \star \\ 0 & \star \end{bmatrix} \ (mod\ N) \right\},$$

of weight k and Dirichlet character χ of eq. (9).

5.4 Properties of $M_k(\Gamma_0(N), \chi)$

The space of modular forms $M_k(\Gamma_0(N), \chi)$ is the sum of a subspace of Eisenstein series and of cusp forms

$$M_k(\Gamma_0(N), \chi) = \mathcal{E}_k(\Gamma_0(N), \chi) \oplus \mathcal{S}_k(\Gamma_0(N), \chi). \tag{10}$$

But here, the dimension of the space of Eisenstein series can be more than 1. So, from (10), we deduce that

$$\boxed{\Theta_\Lambda(q) = \Theta_{\Lambda,e}(q) + \Theta_{\Lambda,s}(q)} \tag{11}$$

where $\Theta_{\Lambda,e}(q)$ is a linear combination of Eisenstein series and $\Theta_{\Lambda,s}(q)$ is a cusp form.

5.5 Example in Dimension 4

D_4 is a lattice of level $N = 2$. Gram matrix is

$$\boldsymbol{F} = \begin{bmatrix} 2 & 0 & 1 & 0 \\ 0 & 2 & -1 & 0 \\ 1 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix}$$

and $\det \boldsymbol{F} = 4$. Take $a = \prod p^{\alpha(p)}$. Then, the Dirichlet character is

$$\chi(a) = \left(\frac{4}{a}\right) = \begin{cases} 0 & \text{if } a \equiv 0 \pmod 2 \\ 1 & \text{if } a \equiv 1 \pmod 2 \end{cases}.$$

The congruence modular group defining the modular form where $\Theta_{D_4}(q)$ lives is $\Gamma_0(2)$. Weight of the modular form is $k = 2$. There is, in $M_2(2, \chi)$ one Eisenstein series $E_{2,0}(q)$ and no cusp form. So,

$$\Theta_{D_4}(q) = E_{2,0}(q) = 1 + 24q^2 + 24q^4 + 96q^6 + \cdots$$

Figure 6 gives the secrecy function of D_4. Note that the secrecy gain is now achieved at $y_0 = \frac{1}{\sqrt{2}}$.

5.6 Asymptotic Analysis

For a lattice Λ, whose theta series is given in eq. (11), the series in the Eisenstein subspace, $\Theta_{\Lambda,e}(q)$ is called the singular series. It only depends on the genus of the lattice Λ. When n is large enough, then, the coefficients of $\Theta_{\Lambda,e}(q)$ are asymptotic estimates of the coefficients of the theta series of Λ since they are of a larger order of magnitude than those of the cusp form $\Theta_{\Lambda,s}(q)$. So, there is a concentration result which says that, for n large enough, all lattices in the same genus have a theta series approximately given by $\Theta_{\Lambda,e}(q)$.

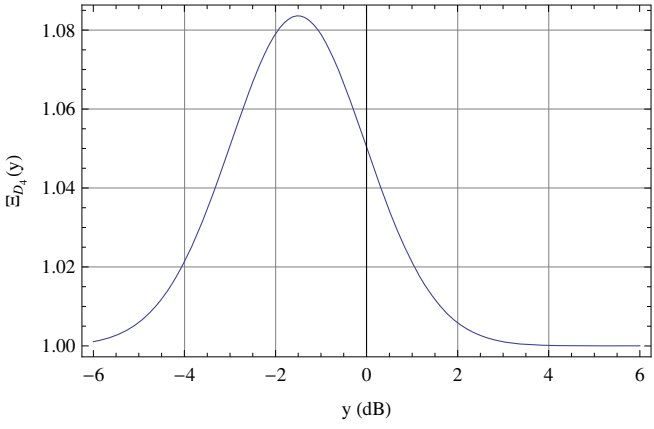


Fig. 6. Secrecy function of the checkerboard lattice D_4

6 Conclusion

The secrecy gain introduced in [8] is a new lattice invariant that measures how much confusion the eavesdropper will experience. This parameter is based on the value of the theta series of lattice Λ_e at some point that depends on the lattice itself. We can analyze how secrecy gain behaves, when dimension grows up, It depends on some Eisenstein series calles the singular series. Next step consists now in studying the behavior of this series and relating these parameters to the system parameters.

References

1. Ozarow, L.H., Wyner, A.D.: Wire-tap channel II. Bell Syst. Tech. Journal 63(10), 2135–2157 (1984)
2. Zamir, R., Shamai, S., Erez, U.: Linear/lattice codes for structured multi-terminal binning. IEEE Trans. Inf. Theory (June 2002)
3. Pradhan, S., Ramchandran, K.: Generalized coset codes for distributed binning. IEEE Trans. Inf. Theory (October 2005)
4. Thangaraj, A., Dihidar, S., Calderbank, A.R., McLaughlin, S., Merolla, J.-M.: Applications of LDPC codes to the wiretap channel. IEEE Trans. Inf. Theory 53(8) (August 2007)
5. Klinc, D., Ha, J., McLaughlin, S., Barros, J., Kwak, B.: LDPC codes for the Gaussian wiretap channel. In: Proc. Information Theory Workshop (October 2009)
6. El Rouayheb, S.Y., Soljanin, E.: On wiretap networks II. In: Proceedings ISIT (2007)
7. He, X., Yener, A.: Providing secrecy with structured codes: Tools and applications to two-user Gaussian channels (July 2009), `http://arxiv.org/pdf/0907.5388`
8. Belfiore, J.-C., Oggier, F.: Secrecy gain: a wiretap lattice code design. In: ISITA 2010 (2010), arXiv:1004.4075v1 [cs.IT]
9. Wyner, A.: The wire-tap channel. Bell. Syst. Tech. Journal 54 (October 1975)

10. Conway, J., Sloane, N.: Sphere packings, Lattices and Groups, 3rd edn. Springer, Heidelberg (1998)
11. Weisstein, E.: Jacobi Theta Functions. MathWorld – A Wolfram Web Resource, http://mathworld.wolfram.com/JacobiThetaFunctions.html
12. Skoruppa, N.-P.: Reduction mod ℓ of Theta Series of level ℓ^n, arXiv:0807.4694v2[math.NT]
13. Ebeling, W.: Lattices and Codes. Advanced Lectures in Mathematics (1994)
14. Serre, J.-P.: A course in arithmetic. Graduate Texts in Mathematics (1996)
15. Oggier, F., Solé, P., Belfiore, J.-C.: Lattice codes for the wiretap gaussian channel: Construction and analysis. IEEE Trans. Inf. Theory (2011) (submitted, on line on ArXiv)
16. Stein, W.: Modular Forms, a Computational Approach. Graduate Studies in Mathematics, vol. 79. AMS (2007)
17. Cassels, J.: Rational Quadratic Forms. Dover Books on Mathematics. Dover, New york (1978)

List Decoding for Binary Goppa Codes*

Daniel J. Bernstein

Department of Computer Science
University of Illinois at Chicago, Chicago, IL 60607–7045, USA
djb@cr.yp.to

Abstract. This paper presents a Patterson-style list-decoding algorithm for classical irreducible binary Goppa codes. The algorithm corrects, in polynomial time, approximately $n - \sqrt{n(n-2t-2)}$ errors in a length-n classical irreducible degree-t binary Goppa code. Compared to the best previous polynomial-time list-decoding algorithms for the same codes, the new algorithm corrects approximately $t^2/2n$ extra errors.

1 Introduction

Patterson in [27] introduced a polynomial-time decoding algorithm that corrects t errors in a classical irreducible degree-t binary Goppa code.

This paper introduces a considerably more complicated, but still polynomial-time, list-decoding algorithm for classical irreducible binary Goppa codes. The advantage of the new algorithm is that it corrects approximately

$$n - \sqrt{n(n-2t-2)} \approx t + 1 + \frac{(t+1)^2}{2(n-t-1)}$$

errors in a length-n degree-t code. Typically t is chosen close to $n/(2 \lg n)$; then the new algorithm corrects approximately $t + 1 + n/(8(\lg n)^2)$ errors.

Comparison to previous list-decoding algorithms. A different strategy for decoding a degree-t classical binary Goppa code $\Gamma_2(\ldots, g)$ is to view it as an "alternant code," i.e., a subfield code of a degree-t generalized Reed–Solomon code $\Gamma_{2^m}(\ldots, g)$ over the larger field $\mathbf{F}_{2^m}$. The generalized Reed–Solomon code can be decoded by several different algorithms: Berlekamp's algorithm, for example, or the famous algorithm introduced by Guruswami and Sudan in [17].

Both of those algorithms are much less effective than Patterson's algorithm. Berlekamp corrects only $t/2$ errors; Guruswami and Sudan correct approximately $n - \sqrt{n(n-t)} \approx t/2 + t^2/(8n)$ errors but still do not reach t errors. Guruswami and Sudan in [17, Section 3.1] point out this limitation of their algorithm ("performance can only be compared with the designed distance, rather than the

* Permanent ID of this document: 210ecf064c479a278ab2c98c379f72e0. Date of this document: 2011.03.02. This work was carried out while the author was visiting Technische Universiteit Eindhoven. This work has been supported in part by the National Science Foundation under grant ITR–0716498 and in part by the Cisco University Research Program.

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 62–80, 2011.
© Springer-Verlag Berlin Heidelberg 2011

actual distance"); they do not mention how serious this limitation is in the binary-Goppa case, where moving to a larger field chops the distance *in half*.

Koetter and Vardy pointed out in 2000 an improvement to the Guruswami–Sudan algorithm for alternant codes defined over small fields. The Koetter–Vardy algorithm corrects approximately $n' - \sqrt{n'(n'-t)}$ errors for a length-n degree-t alternant code over $\mathbf{F}_q$, where $n' = n(q-1)/q$; see, e.g., [18, Section 6.3.8]. In particular, it corrects approximately $(1/2)(n - \sqrt{n(n-2t)}) \approx t/2 + t^2/(4n)$ errors for a length-n degree-t alternant code over $\mathbf{F}_2$, such as a length-n degree-t Goppa code $\Gamma_2(\ldots, g)$. The algorithm still does not reach t errors.

The first draft of this paper, posted in July 2008, appears to have been the first improvement in more than thirty years on the decoding power of Patterson's algorithm. See below for a discussion of followup work. I have not attempted to compare my algorithm to Wu's earlier algorithm [29] for a smaller family of codes, namely narrow-sense binary BCH codes.

Extra errors by brute force. Another standard way to correct extra errors is to guess the positions of the extra errors. For example, one can guess e error positions, flip those e bits, and then apply Patterson's algorithm to correct t additional errors, overall correcting $t+e$ errors. The guess is correct with probability $\binom{n-e}{t}/\binom{n}{t+e}$, so after $\binom{n}{t+e}/\binom{n-e}{t}$ guesses one has a good chance of finding any particular codeword at distance $t+e$. One can bring the chance exponentially close to 1 by moderately increasing the number of guesses.

Although this algorithm involves many repetitions of Patterson's algorithm, it remains a polynomial-time algorithm if e is chosen so that $\binom{n}{t+e}/\binom{n-e}{t}$ grows polynomially. In particular, in the typical case $t \in \Theta(n/\lg n)$, one can decode $\Theta((\lg n)/(\lg \lg n))$ extra errors in polynomial time.

Similarly, one can guess e error positions, flip those e bits, and then apply this paper's new list-decoding algorithm. Compared to Patterson's original algorithm, this method decodes

- the same t errors, plus
- approximately $n - t - \sqrt{n(n-2t-2)}$ extra errors from the new algorithm, plus
- e additional errors from guessing;

and the method remains polynomial-time if e is small. In particular, for $t \approx n/(2\lg n)$, the new algorithm adds approximately $n/(8\lg n)^2$ extra errors, and guessing adds $\Theta((\lg n)/(\lg \lg n))$ extra errors, still in polynomial time.

A refined algorithm analysis would consider the number of errors correctable in time $n^{1+o(1)}$, the number of errors correctable in time $n^{2+o(1)}$, etc., rather than simply the number of errors correctable in polynomial time. This paper does not carry out this level of analysis, and does not incorporate various speedups visible at this level of analysis. Such speedups are evident at several levels of algorithm design: for example, one should use essentially-linear-time FFT-based algorithms for multiplication, multipoint polynomial evaluation, interpolation, etc., rather than quadratic-time schoolbook algorithms. For a survey of the FFT-based algorithms see, e.g., my paper [4].

Comparison to subsequent list-decoding algorithms. Here is another strategy for decoding "wild Goppa codes" $\Gamma_q(\ldots,g^{q-1})$, where g is squarefree: first apply the identity $\Gamma_q(\ldots,g^{q-1}) = \Gamma_q(\ldots,g^q)$ published by Sugiyama, Kasahara, Hirasawa, and Namekawa in [28]; then apply an alternant decoder to $\Gamma_q(\ldots,g^q)$. Note that all squarefree binary Goppa codes $\Gamma_2(\ldots,g)$ are wild Goppa codes.

There are again several choices of alternant decoders. Berlekamp's algorithm decodes $qt/2$ errors in $\Gamma_q(\ldots,g^q)$; combining Berlekamp's algorithm with the SKHN identity decodes $qt/2$ errors in $\Gamma_q(\ldots,g^{q-1})$, and in particular t errors in $\Gamma_2(\ldots,g)$. This algorithm is almost as old as Patterson's algorithm (see [28]), is somewhat simpler than Patterson's algorithm, and corrects the same number of errors. On the other hand, Patterson's algorithm seems to be faster, since it works modulo g rather than modulo g^2, and appears to have become the standard decoding algorithm for these codes. I am not aware of any serious investigation of the speed of the g^2 approach.

Switching from SKHN+Berlekamp to SKHN+Guruswami–Sudan decodes approximately $n - \sqrt{n(n-qt)}$ errors for $\Gamma_q(\ldots,g^{q-1})$, and in particular $n - \sqrt{n(n-2t)}$ errors for $\Gamma_2(\ldots,g)$, just like the algorithm in this paper. The first publication of this SKHN+Guruswami–Sudan list-decoding algorithm, as far as I know, was in the paper [7, Section 5] in July 2010, two years after the first draft of this paper was posted. Compared to my Patterson-style list-decoding algorithm, this list-decoding algorithm has the advantage of simplicity, and the further advantage of generalizing immediately from $\mathbf{F}_2$ to $\mathbf{F}_q$. On the other hand, I would guess that my Patterson-style algorithm is faster, justifying its additional complexity.

SKHN+Koetter–Vardy decodes more errors: approximately $n' - \sqrt{n'(n'-qt)}$ errors for $\Gamma_q(\ldots,g^{q-1})$ where $n' = n(q-1)/q$, and in particular approximately $(1/2)(n - \sqrt{n(n-4t)}) \approx t + t^2/n$ errors for $\Gamma_2(\ldots,g)$. This was pointed out by Augot, Barbier, and Couvreur in [2] in December 2010. However, I would again guess that this paper's technique is faster.

An application to code-based cryptography. McEliece in [24] proposed a public-key encryption system using exactly the same codes considered in this paper. The public key is a generator matrix (or, as proposed by Niederreiter in [26], a parity-check matrix) of a code equivalent to a classical irreducible degree-t binary Goppa code chosen secretly by the receiver. The sender encodes a message and adds t errors; the receiver decodes the errors.

Adding more errors makes McEliece's system harder to break by all known attacks, but also requires the receiver to decode the additional errors, posing the problem tackled in this paper: exactly how many errors can be efficiently decoded in a classical irreducible binary Goppa code? See [6] for further discussion and security analysis. One could also switch to a different class of codes over $\mathbf{F}_2$, but I am not aware of codes over $\mathbf{F}_2$ that allow efficient decoding of more errors!

2 Review of Divisors in Arithmetic Progressions

Consider the problem of finding all divisors of n congruent to u modulo v, where u, v, n are positive integers with $\gcd\{v, n\} = 1$. (What does this have to do with list decoding? Bear with me.)

There is no difficulty if $v \geq n^{1/2}$. Lenstra in [22] published a polynomial-time algorithm for $v \geq n^{1/3}$. Konyagin and Pomerance in [20] published a polynomial-time algorithm for $v \geq n^{3/10}$. Coppersmith, Howgrave-Graham, and Nagaraj found a polynomial-time algorithm for $v \geq n^{\alpha^2}$ for any fixed $\alpha > 1/2$; see [19, Section 5.5] and [15]. (Lenstra subsequently pointed out that one could handle $\alpha = 1/2$, but this extra refinement is not relevant here.) More generally, the Coppersmith–Howgrave-Graham–Nagaraj algorithm finds all divisors of n in an arithmetic progression $u - Hv, u - (H-1)v, \ldots, u - v, u, u + v, \ldots, u + (H-1)v, u + Hv$. The algorithm is polynomial-time if the smallest entry $u - vH$ is $n^{1/\alpha}$ and the number $2H + 1$ of entries is smaller than approximately n^{1/α^2}.

The algorithm actually does more: it finds all small integers s such that the fraction $(s + w)/n$ has small denominator. Here w is the quotient of u by v modulo n. Note that $(s + w)/n$ has denominator at most $n/(u + sv)$ if $u + sv$ divides n: indeed, $(s + w)/n = \bar{v}(u + sv)/n + (w - u\bar{v})/n + s(1 - v\bar{v})/n$, where $\bar{v}$ is the reciprocal of v modulo n.

Boneh later pointed out—see [8]—that the same algorithm can be viewed as a state-of-the-art list-decoding algorithm for "CRT codes" under a standard weighted distance. Take n to be a product of many small primes $p_1, p_2, \ldots$, and consider codewords $(s \bmod p_1, s \bmod p_2, \ldots)$ where $s \in \{-H, \ldots, 0, 1, \ldots, H\}$. A word $(w \bmod p_1, w \bmod p_2, \ldots)$ is close to a codeword $(s \bmod p_1, s \bmod p_2, \ldots)$ if and only if $s - w$ has a large factor in common with n, i.e., $(s - w)/n$ has small denominator.

The algorithm. Fix positive integers ℓ, k with $\ell > k$. Define $L \subset \mathbf{Q}[z]$ as the ℓ-dimensional lattice generated by the polynomials

$$1, \frac{Hz + w}{n}, \left(\frac{Hz + w}{n}\right)^2, \ldots, \left(\frac{Hz + w}{n}\right)^k,$$
$$Hz\left(\frac{Hz + w}{n}\right)^k, (Hz)^2\left(\frac{Hz + w}{n}\right)^k, \ldots, (Hz)^{\ell-k-1}\left(\frac{Hz + w}{n}\right)^k.$$

The Coppersmith–Howgrave-Graham–Nagaraj algorithm uses lattice-basis reduction to find a nonzero vector $\varphi \in L$ with small coefficients. It then finds the desired integers s by finding rational roots s/H of φ.

Specifically, L has determinant $H^{\ell(\ell-1)/2}/n^{\ell k - k(k+1)/2}$, so the well-known LLL algorithm finds φ with norm at most $(2H)^{(\ell-1)/2}/n^{k-k(k+1)/2\ell}$. If $|s/H| \leq 1$ then $\varphi(s/H) \leq \sqrt{\ell}(2H)^{(\ell-1)/2}/n^{k-k(k+1)/2\ell}$; but $\varphi(s/H)$ is also a multiple of $1/D^k$ where D is the denominator of $(s + w)/n$. In particular, $\varphi(s/H)$ must be 0 if $1/D^k > \sqrt{\ell}(2H)^{(\ell-1)/2}/n^{k-k(k+1)/2\ell}$.

The algorithm thus finds all integers $s \in \{-H, \ldots, -1, 0, 1, \ldots, H\}$ such that the denominator of $(s + w)/n$ is smaller than $n^{1-(k+1)/2\ell}/\ell^{1/2k}(2H)^{(\ell-1)/2k}$. By

choosing a moderately large k, and choosing $\ell \approx k\sqrt{(\lg 2n)/\lg 2H}$, one can push the denominator bound up to approximately $n/2^{\sqrt{(\lg 2n)(\lg 2H)}}$, and in particular find divisors larger than approximately $2^{\sqrt{(\lg 2n)(\lg 2H)}}$.

The function-field analogue. The integers in the Coppersmith–Howgrave-Graham–Nagaraj algorithm can be replaced by polynomials over a finite field. The LLL algorithm for integer lattice-basis reduction is replaced by simpler algorithms for polynomial lattice-basis reduction. See, e.g., [21, Section 1] for a gentle introduction to polynomial lattice-basis reduction, or [25, Section 2] for a faster algorithm.

Many of the cryptanalytic applications of the algorithm are uninteresting for polynomials, since polynomials can be factored efficiently into irreducibles. However, the list-decoding application remains interesting for polynomials—it is essentially the Guruswami–Sudan algorithm!

Section 3 extends the Coppersmith–Howgrave-Graham–Nagaraj algorithm to solve a slightly more complicated "linear combinations as divisors" problem. Section 7 presents this paper's new list-decoding method for binary Goppa codes, combining the function-field analogue of the "linear combinations as divisors" algorithm with the extension of Patterson's algorithm presented in Section 6.

3 Linear Combinations as Divisors

The Coppersmith–Howgrave-Graham–Nagaraj algorithm discussed in Section 2, given positive integers u, v, n with $\gcd\{v, n\} = 1$, finds all small integers s such that $u + sv$ divides n. This section explains, more generally, how to find all pairs of small coprime integers (r, s) with $r > 0$ such that $ru + sv$ divides n. The precise meaning of "small" is defined below.

The algorithm sometimes outputs additional pairs (r, s). It is up to the user to check which of the pairs (r, s) is small, has $ru + sv$ dividing n, etc. However, the algorithm is guaranteed to finish quickly (and therefore to output very few pairs), and its output is guaranteed to include all of the desired pairs (r, s).

The algorithm. Compute the quotient w of u by v modulo n. This algorithm actually looks for small coprime (r, s) such that $(s + rw)/n$ has small denominator.

Fix positive integers G, H, and define $\Theta = H/G$. The algorithm focuses on pairs (r, s) such that $1 \le r \le G$ and $-H \le s \le H$.

Fix positive integers ℓ, k with $\ell > k$. Define $L \subset \mathbf{Q}[z]$ as the ℓ-dimensional lattice generated by the polynomials

$$1, \frac{\Theta z + w}{n}, \left(\frac{\Theta z + w}{n}\right)^2, \ldots, \left(\frac{\Theta z + w}{n}\right)^k,$$
$$\Theta z \left(\frac{\Theta z + w}{n}\right)^k, (\Theta z)^2 \left(\frac{\Theta z + w}{n}\right)^k, \ldots, (\Theta z)^{\ell-k-1} \left(\frac{\Theta z + w}{n}\right)^k.$$

Use lattice-basis reduction to find a nonzero vector $\varphi \in L$ with small coefficients.

For each rational root of φ: Multiply the root by Θ, write the product in the form s/r with $\gcd\{r,s\} = 1$ and $r > 0$, and output (r,s).

What the algorithm accomplishes. Observe that the determinant of L is $\Theta^{\ell(\ell-1)/2}/n^{\ell k-k(k+1)/2}$. Reduction guarantees that

$$\sqrt{\varphi_0^2+\varphi_1^2+\cdots} \le \frac{(2\Theta)^{(\ell-1)/2}}{n^{k-k(k+1)/2\ell}}.$$

Assume that $1 \le r \le G$ and $-H \le s \le H$. Then

$$\begin{aligned}\left|r^{\ell-1}\varphi\left(\frac{s}{\Theta r}\right)\right| &= \left|\varphi_0 r^{\ell-1}+\varphi_1 r^{\ell-2}\frac{s}{\Theta}+\cdots+\varphi_{\ell-1}\frac{s^{\ell-1}}{\Theta^{\ell-1}}\right| \\ &\le \sqrt{(r^{\ell-1})^2+\cdots+\left(\frac{s^{\ell-1}}{\Theta^{\ell-1}}\right)^2}\sqrt{\varphi_0^2+\varphi_1^2+\cdots} \\ &\le \sqrt{\ell}G^{\ell-1}\frac{(2\Theta)^{(\ell-1)/2}}{n^{k-k(k+1)/2\ell}} = \frac{\sqrt{\ell}(2GH)^{(\ell-1)/2}}{n^{k-k(k+1)/2\ell}}.\end{aligned}$$

Assume further that $(s+rw)/n$ has denominator D. Then $(s/r+w)/n$ is a multiple of $1/Dr$ so all of

$$1, \frac{s/r+w}{n}, \ldots, \left(\frac{(s/r+w)}{n}\right)^k, \ldots, (s/r)^{\ell-k-1}\left(\frac{(s/r+w)}{n}\right)^k$$

are multiples of $(1/r)^{\ell-k-1}(1/Dr)^k = 1/D^k r^{\ell-1}$. Thus $\varphi(s/\Theta r)$ is a multiple of $1/D^k r^{\ell-1}$; i.e., $r^{\ell-1}\varphi(s/\Theta r)$ is a multiple of $1/D^k$.

Now assume additionally that $D < n^{1-(k+1)/2\ell}/\ell^{1/2k}(2GH)^{(\ell-1)/2k}$. Then $1/D^k > \sqrt{\ell}(2GH)^{(\ell-1)/2}/n^{k-k(k+1)/2\ell}$, so $r^{\ell-1}\varphi(s/\Theta r)$ must be 0; i.e., $s/\Theta r$ is a root of φ. The algorithm finds $s/\Theta r$ if $\gcd\{r,s\} = 1$.

In particular, if $ru+sv$ is a divisor of n with $1 \le r \le G$, $-H \le s \le H$, $\gcd\{r,s\} = 1$, and $ru+sv > \ell^{1/2k}(2GH)^{(\ell-1)/2k}n^{(k+1)/2\ell}$, then the algorithm outputs (r,s).

By choosing a moderately large k, and choosing $\ell \approx k\sqrt{(\lg 2n)/\lg 2GH}$, one can push the bound $\ell^{1/2k}(2GH)^{(\ell-1)/2k}n^{(k+1)/2\ell}$ down to approximately $2^{\sqrt{(\lg 2n)(\lg 2GH)}}$.

Comparison to other "Coppersmith-type" algorithms. My survey paper [5] discusses a general method that, given a polynomial f, finds all small-height rational numbers s/r such that $f(s/r)$ has small height. Here "small height" means "small numerator and small denominator." This includes finding divisors in residue classes and finding codeword errors beyond half the minimum distance, as discussed in Section 2; other standard applications are finding divisors in short intervals, finding high-power divisors, and finding modular roots. See [5] for credits and historical discussion, including both the rational-number-field case and the rational-function-field case.

All of the applications mentioned in the previous paragraph specify the denominator r; in other words, they find all small *integers* s such that $f(s)$ has

small height. But this limitation is not inherent in the method. The method discovers small pairs (r, s) even if both r and s are allowed to vary.

(Recently Cohn and Heninger in [13] have generalized the method to cover all global fields—but with the same limitation, searching only for integral elements of those fields. Presumably the limitation can be removed.)

In particular, one can efficiently find all small-height rational numbers s/r such that $(s/r + w)/n$ has small height—in particular, all small-height rational numbers s/r such that $ru + sv$ divides n. What I have shown in this section is that, for divisors $ru + sv \approx n^{1/\alpha}$, "small" includes all (r, s) with rs up to approximately n^{1/α^2}.

The same method generalizes to polynomials f in more variables. One can, for example, find all small integer pairs (r, s) such that $f(r, s)$ has small height. However, the bivariate method is considerably more difficult to analyze and optimize than the univariate method. Even when the bivariate method can be proven to work, it typically searches fewer f inputs than the univariate method. What the algorithm in this section illustrates is that *homogeneous* bivariate polynomials are almost as easy to handle as univariate polynomials.

In [12]—more than a year after I posted the first draft of this paper and more than five years after I posted the first draft of [5]—Castagnos, Joux, Laguillaumie, and Nguyen published a "new *rigorous* homogeneous bivariate variant of Coppersmith's method." They used this method to attack a cryptosystem. Their "new" variant is, in fact, an uncredited special case of the method in [5]—the same special case that I had, for the same reasons, already highlighted in this paper.

The function-field analogue. The integers in this section's $ru+sv$ algorithm, like the integers in other Coppersmith-type algorithms, can be replaced by polynomials over a finite field. The resulting algorithm can be used for list decoding of classical irreducible binary Goppa codes. See Section 7.

In this application one cares only about *squares* r, s. In other words, one wants to find divisors of n of the form $r^2u + s^2v$. One can apply lattice-basis-reduction methods directly to the polynomial $(s^2 + r^2w)/n$, but I don't see how this would allow larger rs. Perhaps I'm missing an easy factor-of-2 improvement (in general, or in the characteristic-2 case), or perhaps there's an explanation for why this type of improvement can't work.

4 Review of Classical Irreducible Binary Goppa Codes

This section reviews three equivalent definitions of the classical irreducible binary Goppa code $\Gamma_2(a_1, \ldots, a_n, g)$: the "polynomial" definition, the "classical" definition, and the "evaluation" definition.

The notations $m, t, n, a_1, \ldots, a_n, g, h, \Gamma$ in this section will be reused in Sections 5, 6, and 7.

Parameters for the code. Fix an integer $m \geq 3$. Typically $m \in \{10, 11, 12\}$ in the cryptographic applications mentioned in Section 1.

Fix an integer t with $2 \leq t \leq (2^m - 1)/m$. The Goppa code will be a "degree-t code" designed to correct t errors. Extremely small and extremely large values of t are not useful, but intermediate values of t produce interesting codes; for $m = 11$ one could reasonably take (e.g.) $t = 32$, or $t = 70$, or $t = 100$.

Fix an integer n with $mt + 1 \leq n \leq 2^m$. It is common to restrict attention to the extreme case $n = 2^m$; e.g., $n = 2048$ if $m = 11$. However, a wider range of n allows a better security/efficiency tradeoff for code-based cryptography, as illustrated in [6, Section 7] and [7].

Fix a sequence $a_1, \ldots, a_n$ of distinct elements of the finite field $\mathbf{F}_{2^m}$. Typically $n = 2^m$ and $a_1, \ldots, a_n$ are chosen as all the elements of $\mathbf{F}_{2^m}$ in lexicographic order, given a standard basis for $\mathbf{F}_{2^m}$ over $\mathbf{F}_2$. For $n < 2^m$ there is more flexibility.

Finally, fix a monic degree-t irreducible polynomial $g \in \mathbf{F}_{2^m}[x]$. There are no standard choices here; in the classic study of minimum distance it is an open problem to find the best g, and in code-based cryptography it is important for g to be a randomly chosen secret.

The "polynomial" view of the code. Define $h = \prod_i (x - a_i) \in \mathbf{F}_{2^m}[x]$. In the extreme case $n = 2^m$, this polynomial h is simply $x^n - x$, with derivative $h' = nx^{n-1} - 1 = 1$, slightly simplifying some of the formulas below.

Define

$$\Gamma = \Gamma_2(a_1, \ldots, a_n, g) = \left\{ c \in \mathbf{F}_2^n : \sum_i c_i \frac{h}{x - a_i} \bmod g = 0 \right\}.$$

This set Γ is the kernel of the "syndrome" map $\mathbf{F}_2^n \to \mathbf{F}_{2^m}^t$ that maps c to the coefficients of $1, x, \ldots, x^{t-1}$ in $\sum_i c_i h/(x - a_i) \bmod g$; consequently Γ is an $\mathbf{F}_2$-module of dimension at least $n - mt$, i.e., an $[n, \geq n - mt]$ code over $\mathbf{F}_2$.

In other words: The polynomials $h/(x - a_1) \bmod g$, $h/(x - a_2) \bmod g$, $\ldots$, $h/(x - a_n) \bmod g$, viewed as vectors over $\mathbf{F}_2$, form a parity-check matrix for the code Γ.

The "classical" view of the code. By construction g has degree $t \geq 2$, and has none of $a_1, \ldots, a_n$ as roots. Therefore h is coprime to g.

Consequently the polynomial $\sum_i c_i h/(x - a_i)$ in $\mathbf{F}_{2^m}[x]$ is a multiple of g if and only if $\sum_i c_i/(x - a_i)$ equals 0 in the field $\mathbf{F}_{2^m}[x]/g$. The classical Goppa code associated to $a_1, \ldots, a_n, g$ is most commonly defined as the set of $c \in \mathbf{F}_2^n$ such that $\sum_i c_i/(x - a_i) = 0$ in $\mathbf{F}_{2^m}[x]/g$; this is the same code as Γ.

Another consequence of the coprimality of h and g is that the minimum distance of Γ is at least $2t + 1$; i.e., Γ is an $[n, \geq n - mt, \geq 2t + 1]$ code over $\mathbf{F}_2$. Proof: If $c \in \Gamma - \{0\}$ then g divides the polynomial $\sum_i c_i h/(x - a_i) = \sum_{i:c_i=1} h/(x - a_i) = h\epsilon'/\epsilon$ where $\epsilon = \prod_{i:c_i=1}(x - a_i)$. Thus g divides ϵ'. Write ϵ as $\alpha^2 + x\beta^2$, and observe that $\beta \neq 0$, since by construction ϵ is not a square. Now $\epsilon' = \beta^2$, so g divides β^2; but g is irreducible, so g divides β, so β has degree at least t, so ϵ has degree at least $2t + 1$.

The "evaluation" view of the code. Define

$$\Gamma_{2^m}(a_1, \ldots, a_n, g) = \left\{ c \in \mathbf{F}_{2^m}^n : \sum_i c_i \frac{h}{x - a_i} \bmod g = 0 \right\}.$$

This set $\Gamma_{2^m}(a_1, \ldots, a_n, g)$ is an $[n, n-t]$ code over $\mathbf{F}_{2^m}$. The classical binary Goppa code Γ is a subfield code of $\Gamma_{2^m}(a_1, \ldots, a_n, g)$.

If f is a polynomial in $\mathbf{F}_{2^m}[x]$ with $\deg f < n - t$ then the vector

$$(f(a_1)g(a_1)/h'(a_1), f(a_2)g(a_2)/h'(a_2), ..., f(a_n)g(a_n)/h'(a_n))$$

is in $\Gamma_{2^m}(a_1, \ldots, a_n, g)$. Indeed, $\sum_i (f(a_i)g(a_i)/h'(a_i))h/(x - a_i) = fg$ by Lagrange interpolation, and $fg \bmod g = 0$.

Conversely, every element of $\Gamma_{2^m}(a_1, \ldots, a_n, g)$ can be written as a vector of this form: if $\sum_i c_i h/(x-a_i) \in \mathbf{F}_{2^m}[x]$ is a multiple of g, say fg, then $f(a_i)g(a_i) = c_i h'(a_i)$ so $c = (f(a_1)g(a_1)/h'(a_1), f(a_2)g(a_2)/h'(a_2), ..., f(a_n)g(a_n)/h'(a_n))$.

Therefore $\Gamma_{2^m}(a_1, \ldots, a_n, g)$ is a geometric Goppa code, specifically a genus-0 geometric Goppa code, specifically a geometric Goppa code over the projective line.

5 Review of Patterson's Algorithm

This section reviews Patterson's algorithm for correcting t (or fewer) errors in the classical irreducible binary Goppa code $\Gamma = \Gamma_2(a_1, \ldots, a_n, g)$ defined in Section 4.

The algorithm. The input to the algorithm is a vector $w \in \mathbf{F}_2^n$. The output is a list of all codewords $c \in \Gamma$ such that the Hamming distance $|c - w| = \#\{i : c_i \neq w_i\}$ is at most t. There is at most one such codeword—recall that the minimum distance of Γ is at least $2t + 1$.

Define the **norm** $|\varphi|$ of a polynomial $\varphi \in \mathbf{F}_{2^m}[x]$ as $2^{\deg \varphi}$ if $\varphi \neq 0$ and 0 if $\varphi = 0$. Extend the norm multiplicatively to rational functions $\varphi \in \mathbf{F}_{2^m}(x)$: the norm $|\varphi/\psi|$ is $|\varphi|/|\psi|$. For example, $|x^3/(x^5 + x + 1)| = |x^3|/|x^5 + x + 1| = 2^3/2^5 = 2^{-2}$.

Compute the square root of $(\sum_i w_i/(x-a_i))^{-1} - x$ in the field $\mathbf{F}_{2^m}[x]/g$. This computation fails if $\sum_i w_i/(x - a_i)$ is zero in the field; if so, output w and stop.

Lift the square root to a polynomial $s \in \mathbf{F}_{2^m}[x]$ of degree $< t$. Apply lattice-basis reduction to the lattice $L \subseteq \mathbf{F}_{2^m}[x]^2$ generated by the vectors $(s, 1)$ and $(g, 0)$, obtaining a minimum-length nonzero vector (α_0, β_0). Here the length $|(\alpha, \beta)|$ of a vector $(\alpha, \beta) \in \mathbf{F}_{2^m}[x]^2$ is, by definition, the norm of the polynomial $\alpha^2 + x\beta^2$.

Compute $\epsilon_0 = \alpha_0^2 + x\beta_0^2$. Use a polynomial-factorization algorithm to see whether the monic part of ϵ_0 (i.e., ϵ_0 divided by its leading coefficient) splits into distinct linear factors of the form $x - a_i$. If it does, output the unique vector $c \in \mathbf{F}_2^n$ such that $\{i : w_i \neq c_i\} = \{i : \epsilon_0(a_i) = 0\}$.

Why the algorithm works. If the algorithm outputs w in the first step then $\sum_i w_i/(x - a_i) = 0$ in the field $\mathbf{F}_{2^m}[x]/g$ so $w \in \Gamma$. Conversely, if $w \in \Gamma$ then the algorithm correctly outputs w in the first step. Note that in this case there are no other codewords at distance $\leq 2t$.

Assume from now on that $w \notin \Gamma$. Then $\sum_i w_i/(x - a_i) \neq 0$ in $\mathbf{F}_{2^m}[x]/g$.

The specified basis $(s,1),(g,0)$ of L has orthogonalization $(0,1),(g,0)$, with lengths $|(0,1)| = |x| = 2^1$ and $(g,0) = |g^2| = 2^{2t}$, product 2^{2t+1}. Consequently $|(\alpha_0,\beta_0)| \le 2^{(2t+1)/2} = 2^{t+1/2}$; i.e., $\deg \epsilon_0 \le t + 1/2$; i.e., $\deg \epsilon_0 \le t$.

Furthermore, the lattice L is exactly the set of vectors $(\alpha,\beta) \in \mathbf{F}_{2^m}[x]^2$ such that $\alpha - s\beta$ is a multiple of g. Consequently any $(\alpha,\beta) \in L$ satisfies $\alpha^2/\beta^2 = s^2 = (\sum_i w_i/(x-a_i))^{-1} - x$ in the field $\mathbf{F}_{2^m}[x]/g$, if β is not a multiple of g. The polynomial $\epsilon = \alpha^2 + x\beta^2 \in \mathbf{F}_{2^m}[x]$ satisfies $\epsilon' = \beta^2$, so $\epsilon/\epsilon' = \alpha^2/\beta^2 + x = (\sum_i w_i/(x-a_i))^{-1}$ in the field $\mathbf{F}_{2^m}[x]/g$.

If the algorithm outputs a vector c then the monic part of ϵ_0 splits into linear factors, so ϵ_0 is not a square, so $\alpha_0^2 + x\beta_0^2$ is not a square, so $\beta_0 \neq 0$; but $\deg \beta_0 \le (t-1)/2 < t = \deg g$, so β_0 is not a multiple of g, so $\epsilon_0/\epsilon_0' = (\sum_i w_i/(x-a_i))^{-1}$ in the field $\mathbf{F}_{2^m}[x]/g$. Thus $\sum_i w_i/(x-a_i) = \epsilon_0'/\epsilon_0 = \sum_{i:\epsilon_0(a_i)=0} 1/(x-a_i) = \sum_{i:w_i\neq c_i} 1/(x-a_i) = \sum_i (w_i-c_i)/(x-a_i) = \sum_i w_i/(x-a_i) - \sum_i c_i/(x-a_i)$ in the field $\mathbf{F}_{2^m}[x]/g$. Subtract to see that $\sum_i c_i/(x-a_i) = 0$ in the field $\mathbf{F}_{2^m}[x]/g$, i.e., that $c \in \Gamma$. The Hamming distance $|w-c|$ is exactly $\#\{i : \epsilon_0(a_i) = 0\} = \deg \epsilon_0 \le t$. Summary: The output of the algorithm is a codeword at distance $\le t$ from w.

Conversely, assume that $c \in \Gamma$ has $|w-c| \le t$. Define $\epsilon = \prod_{i:w_i\neq c_i}(x-a_i) \in \mathbf{F}_{2^m}[x]$, and write ϵ in the form $\alpha^2 + x\beta^2$. Then $\sum_i c_i/(x-a_i) = 0$ in $\mathbf{F}_{2^m}[x]/g$, so $\sum_i w_i/(x-a_i) = \sum_i (w_i-c_i)/(x-a_i) = \sum_{i:w_i\neq c_i} 1/(x-a_i) = \epsilon'/\epsilon$ in $\mathbf{F}_{2^m}[x]/g$, so $s^2 = \epsilon/\epsilon' - x = \alpha^2/\beta^2$ in $\mathbf{F}_{2^m}[x]/g$. Squaring in the field $\mathbf{F}_{2^m}[x]/g$ is injective, so $s = \alpha/\beta$ in $\mathbf{F}_{2^m}[x]/g$, so $\alpha - s\beta$ is a multiple of g in $\mathbf{F}_{2^m}[x]$; i.e., $(\alpha,\beta) \in L$. Furthermore $\deg \epsilon \le t$ so $|(\alpha,\beta)| \le 2^t$ so $|(\alpha,\beta)||(\alpha_0,\beta_0)| \le 2^{2t}$. Every basis of L has product of lengths at least $|(0,1)||(g,0)| \ge 2^{2t+1}$, so $(\alpha,\beta),(\alpha_0,\beta_0)$ are not a basis; i.e., (α,β) is parallel to (α_0,β_0); but (α_0,β_0) has minimum length in L, so (α,β) is a multiple of (α_0,β_0), say $q(\alpha_0,\beta_0)$ where $q \in \mathbf{F}_{2^m}[x]$. Now $\epsilon = \alpha^2 + x\beta^2 = q^2(\alpha_0^2 + x\beta_0^2) = q^2\epsilon_0$. By construction ϵ is squarefree so ϵ/ϵ_0 is a constant. Hence the monic part of ϵ_0 splits into exactly the distinct linear factors $x - a_i$ that divide ϵ, and the algorithm finds exactly the codeword c.

Numerical example. Define $m = 8$, $n = 2^m = 256$, and $t = 22$. Construct $\mathbf{F}_{2^m}$ as $\mathbf{F}_2[\zeta]/(\zeta^8 + \zeta^4 + \zeta^3 + \zeta^2 + 1)$. Define $a_1 = \zeta$, $a_2 = \zeta^2$, and so on through $a_{255} = \zeta^{255} = 1$; define $a_{256} = 0$. Choose $g = x^{22} + x^{17} + x^{15} + x^{12} + x^5 + \zeta^{78} \in \mathbf{F}_{2^m}[x]$; one can easily check that g is irreducible.

Now the Goppa code Γ is a $[256, \ge 80, \ge 45]$ code over $\mathbf{F}_2$. I generated a random element of Γ and added 22 random errors to it, obtaining the word

$$w = (0,1,1,0,0,1,0,0,1,0,1,1,1,0,1,0,0,1,1,0,1,1,0,1,0,1,1,0,0,1,1,1,1,$$
$$1,1,1,0,1,0,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,0,0,1,0,1,0,0,0,$$
$$0,1,1,0,0,0,1,0,1,0,1,0,0,0,1,1,1,1,0,1,0,0,0,1,1,0,0,0,1,0,1,1,$$
$$0,1,1,1,0,0,0,1,1,0,0,1,1,1,0,0,1,0,1,1,0,0,1,1,1,1,0,1,1,0,0,0,$$
$$0,0,0,0,1,1,1,0,0,1,1,1,1,0,1,0,0,0,0,0,1,0,0,1,0,1,1,1,1,1,1,0,$$
$$0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,1,1,1,0,0,0,1,1,1,$$
$$0,1,1,1,1,1,1,0,0,0,1,0,0,1,1,1,1,1,0,1,1,0,0,0,0,0,1,0,0,1,1,1,$$
$$1,0,1,1,0,0,0,1,0,0,0,1,1,0,0,0,0,1,0,0,0,1,1,1,0,1,1,1,1,0,0,0)$$

in $\mathbf{F}_2^n$. Here is what Patterson's algorithm does with this word w.

The sum $\sum_i w_i/(x-a_i)$ in the field $\mathbf{F}_{2^m}[x]/g$ is $1/(x-a_2)+1/(x-a_3)+1/(x-a_6)+\cdots = 1/(x-\zeta^2)+1/(x-\zeta^3)+1/(x-\zeta^6)+\cdots = \zeta^{64}+\zeta^{110}x+\zeta^{204}x^2+\zeta^{53}x^3+\zeta^{91}x^4+\zeta^{200}x^5+\zeta^{147}x^6+\zeta^{67}x^7+\zeta^{196}x^8+\zeta^{253}x^9+\zeta^{235}x^{10}+\zeta^{161}x^{11}+\zeta^{92}x^{12}+\zeta^{146}x^{13}+\zeta^{125}x^{14}+\zeta^{141}x^{15}+\zeta^{9}x^{16}+\zeta^{34}x^{17}+\zeta^{15}x^{18}+\zeta^{139}x^{19}+\zeta^{229}x^{20}+\zeta^{68}x^{21}$. Invert, subtract x, and compute a square root, namely $\zeta^{200}+\zeta^{46}x+\zeta^{51}x^2+\zeta^{91}x^3+\zeta^{232}x^4+\zeta^{12}x^5+\zeta^{179}x^6+\zeta^{3}x^7+\zeta^{146}x^8+\zeta^{93}x^9+\zeta^{130}x^{10}+\zeta^{92}x^{11}+\zeta^{28}x^{12}+\zeta^{219}x^{13}+\zeta^{96}x^{14}+\zeta^{114}x^{15}+\zeta^{131}x^{16}+\zeta^{61}x^{17}+\zeta^{251}x^{18}+\zeta^{76}x^{19}+\zeta^{237}x^{20}+\zeta^{40}x^{21}$. Define s as this polynomial in $\mathbf{F}_{2^m}[x]$.

The vector $(g,0)$ has degree $(22,0)$ and therefore length 2^{44}. The vector $(s,1)$ has degree $(21,0)$ and therefore length 2^{42}. The quotient $\lfloor g/s \rfloor$ is $\zeta^{-40}x-\zeta^{237-80}=\zeta^{215}x-\zeta^{157}$; the difference $(g,0)-\lfloor g/s\rfloor(s,1)$ is the vector $(g \bmod s, \zeta^{215}x-\zeta^{157})$, which has degree $(20,1)$ and therefore length 2^{40}. Continued reduction eventually produces the vector (α_0,β_0) where $\alpha_0=\zeta^{181}+\zeta^{216}x+\zeta^{219}x^2+\zeta^{188}x^3+\zeta^{69}x^4+\zeta^{126}x^5+\zeta^{145}x^6+\zeta^{233}x^7+\zeta^{243}x^8+\zeta^{31}x^9+\zeta^{182}x^{10}+x^{11}$ and $\beta_0=\zeta^{105}+\zeta^{50}x+\zeta^{5}x^2+\zeta^{116}x^3+\zeta^{150}x^4+\zeta^{123}x^5+\zeta^{7}x^6+\zeta^{224}x^7+\zeta^{220}x^8+\zeta^{84}x^9+\zeta^{150}x^{10}$; this vector has degree $(11,10)$ and therefore length $2^{22} \le 2^t$.

The polynomial $\epsilon_0=\alpha_0^2+x\beta_0^2$ splits into 22 linear factors, namely $x-\zeta^7$, $x-\zeta^{25}$, $x-\zeta^{51}$, $x-\zeta^{60}$, $x-\zeta^{68}$, $x-\zeta^{85}$, $x-\zeta^{126}$, $x-\zeta^{135}$, $x-\zeta^{136}$, $x-\zeta^{138}$, $x-\zeta^{155}$, $x-\zeta^{167}$, $x-\zeta^{168}$, $x-\zeta^{172}$, $x-\zeta^{173}$, $x-\zeta^{189}$, $x-\zeta^{191}$, $x-\zeta^{209}$, $x-\zeta^{212}$, $x-\zeta^{214}$, $x-\zeta^{234}$, $x-\zeta^{252}$. Consequently w has distance 22 from the codeword $c \in \Gamma$ obtained by correcting positions 7, 25, 51, etc.

6 Extracting More Information from Patterson's Algorithm

If Patterson's algorithm is given a word w at distance more than t from the closest codeword—in other words, if the error polynomial ϵ has degree larger than t—then the algorithm's output is empty. However, a closer look at the same calculations reveals more information about ϵ. This section presents an easy extension of Patterson's algorithm, identifying two polynomials ϵ_0, ϵ_1 such that ϵ is a small linear combination of ϵ_0, ϵ_1.

The algorithm. The input, as before, is a vector $w \in \mathbf{F}_2^n$. Assume that $w \notin \Gamma$.

Compute the square root of $(\sum_i w_i/(x-a_i))^{-1}-x$ in the field $\mathbf{F}_{2^m}[x]/g$, and lift it to a polynomial $s \in \mathbf{F}_{2^m}[x]$ of degree below t.

Apply lattice-basis reduction to the lattice $L \subseteq \mathbf{F}_{2^m}[x]^2$ generated by the vectors $(s,1)$ and $(g,0)$, obtaining a minimum-length nonzero vector (α_0,β_0) *and* a minimum-length independent vector (α_1,β_1). Here the length $|(\alpha,\beta)|$ of a vector $(\alpha,\beta) \in \mathbf{F}_{2^m}[x]^2$ is, as before, the norm of the polynomial $\alpha^2+x\beta^2$.

Compute $\epsilon_0=\alpha_0^2+x\beta_0^2$ and $\epsilon_1=\alpha_1^2+x\beta_1^2$. Output (ϵ_0,ϵ_1).

What the algorithm accomplishes. Reduction guarantees that $|(\alpha_0,\beta_0)| \le 2^{(2t+1)/2}$ and that $|(\alpha_0,\beta_0)||(\alpha_1,\beta_1)| = 2^{2t+1}$. Thus $\deg \epsilon_0 \le t$, as in Section 5, and $\deg \epsilon_0 + \deg \epsilon_1 = 2t+1$.

Fix $c \in \Gamma$. Define $\epsilon = \prod_{i:w_i \neq c_i}(x - a_i) \in \mathbf{F}_{2^m}[x]$, and write ϵ in the form $\alpha^2 + x\beta^2$. Then $(\alpha, \beta) \in L$, exactly as in Section 5, so (α, β) can be written as $q_0(\alpha_0, \beta_0) + q_1(\alpha_1, \beta_1)$ for some polynomials q_0, q_1. Consequently $\epsilon = q_0^2\epsilon_0 + q_1^2\epsilon_1$.

The explicit formulas $q_0 = (\alpha\beta_1 - \beta\alpha_1)/g$ and $q_1 = (\alpha\beta_0 - \beta\alpha_0)/g$ show that q_0 and q_1 are very small if ϵ is small. Specifically, fix an integer $u \geq 0$, and assume that $\deg \epsilon \leq t + u$; also write $t_0 = \deg \epsilon_0$, and note that $\deg \epsilon_1 = 2t + 1 - t_0$. Then $\deg \alpha_0 \leq \lfloor t_0/2 \rfloor$, $\deg \beta_0 \leq \lfloor (t_0 - 1)/2 \rfloor$, $\deg \alpha_1 \leq \lfloor (2t + 1 - t_0)/2 \rfloor$, $\deg \beta_1 \leq \lfloor (2t - t_0)/2 \rfloor$, $\deg \alpha \leq \lfloor (t + u)/2 \rfloor$, and $\deg \beta \leq \lfloor (t + u - 1)/2 \rfloor$, so $\deg q_0 \leq \lfloor (t + u + 2t - t_0)/2 \rfloor - t = \lfloor (t + u - t_0)/2 \rfloor$ and $\deg q_1 \leq \lfloor (t + u + t_0 - 1)/2 \rfloor - t = \lfloor (t_0 + u - t - 1)/2 \rfloor$.

Using the results of the algorithm. In the simplest case $u = 0$ (i.e., $\deg \epsilon \leq t$), the degree of q_1 is at most $\lfloor (t_0 - t - 1)/2 \rfloor \leq \lfloor -1/2 \rfloor < 0$, so $q_1 = 0$, so $\epsilon = q_0^2\epsilon_0$. Evidently constant multiples of ϵ_0 are the only possible squarefree choices for ϵ, and one can simply check whether the monic part of ϵ_0 splits into linear factors. This is exactly what Patterson's algorithm does.

However, for larger u, both q_0 and q_1 can be nonzero, and it is not so easy to see which choices for ϵ are possible. There are $\approx 2^{mu}$ coprime polynomial pairs (q_0, q_1) matching the degree bounds; enumerating all of those pairs is practical for a tiny fixed u, such as $u = 1$, but becomes intolerably slow as u increases.

The main point of this paper is an asymptotically much faster algorithm to pin down the possibilities for ϵ. See Section 7.

Refinement: $\gcd\{\epsilon_1, h\} = 1$. There are many choices of ϵ_1: one can adjust (α_1, β_1), without changing its length, by adding small multiples of (α_0, β_0) to it. In particular, for any $r \in \mathbf{F}_{2^m}$, one can replace (α_1, β_1) by $(\alpha_1, \beta_1) + \sqrt{r}(\alpha_0, \beta_0)$, replacing ϵ_1 by $\epsilon_1 + r\epsilon_0$.

It will be convenient later to choose ϵ_1 coprime to h. In practice it seems that, by trying several $r \in \mathbf{F}_{2^m}$, one easily finds r such that $\epsilon_1 + r\epsilon_0$ is coprime to h; consequently, replacing ϵ_1 with $\epsilon_1 + r\epsilon_0$, one obtains ϵ_1 coprime to h.

Can it be *proven* that this is always possible? Here are some remarks on this topic. I am indebted to Tanja Lange for related discussions, and for helpful comments on other parts of this paper.

If $\epsilon_1 + r_1\epsilon_0$ and $\epsilon_1 + r_2\epsilon_0$, with $r_1 \neq r_2$, have a common root s, then s is also a root of $((\epsilon_1 + r_1\epsilon_0) - (\epsilon_1 + r_2\epsilon_0))/(r_1 - r_2) = \epsilon_0$ and $(r_2(\epsilon_1 + r_1\epsilon_0) - r_1(\epsilon_1 + r_2\epsilon_0))/(r_2 - r_1) = \epsilon_1$, so s is a root of $(\epsilon_0\epsilon_1)' = g^2$, contradicting the irreducibility of g. Consequently each $s \in \mathbf{F}_{2^m}$ is a root of $\epsilon_1 + r\epsilon_0$ for at most one $r \in \mathbf{F}_{2^m}$.

Suppose that, for each $r \in \mathbf{F}_{2^m}$, there is a root $s \in \mathbf{F}_{2^m}$ of $\epsilon_1 + r\epsilon_0$. Counting then shows that each $\epsilon_1 + r\epsilon_0$ has exactly one root s, and that each s is a root of exactly one $\epsilon_1 + r\epsilon_0$. In particular, if $n < 2^m$, then there exists an $s \in \mathbf{F}_{2^m}$ that is not a root of h, and the corresponding $\epsilon_1 + r\epsilon_0$ is coprime to h as desired. The only remaining case is $n = 2^m$.

Fix s, and find the unique r such that s is a root of $\epsilon_1 + r\epsilon_0$. Then $\epsilon_1(s) = r\epsilon_0(s)$. Furthermore $\epsilon_0(s) \neq 0$: otherwise $\epsilon_1(s) = 0$, contradicting the irreducibility of g as above. Consequently $\epsilon_1(s)/\epsilon_0(s) = r$. Therefore the rational function ϵ_1/ϵ_0, applied to $\mathbf{F}_{2^m}$, is a "permutation function": it takes each value in $\mathbf{F}_{2^m}$ exactly once.

Note that a uniform random function $\mathbf{F}_{2^m} \to \mathbf{F}_{2^m}$ has probability only about $\exp(-2^m)$ of being a permutation function: for example, probability about 2^{-369} for $m = 8$. One does not expect to bump into a permutation function by chance! But this heuristic is not a proof. Some simple rational functions—including all linear functions, squares of linear functions, etc.—are permutation functions on $\mathbf{F}_{2^m}$. Is there any reason that ϵ_1/ϵ_0 *cannot* be a permutation function?

Define $\varphi = (\epsilon_0(x)\epsilon_1(y) - \epsilon_1(x)\epsilon_0(y))/(x - y) \in \mathbf{F}_{2^m}[x, y]$. If $s_1 \neq s_2$ then $\epsilon_1(s_1)/\epsilon_0(s_1) \neq \epsilon_1(s_2)/\epsilon_0(s_2)$ so $\varphi(s_1, s_2) \neq 0$. Furthermore $\varphi(x, x) = \epsilon_0\epsilon_1' - \epsilon_1\epsilon_0' = (\epsilon_0\epsilon_1)' = g^2$; therefore $\varphi(s, s) \neq 0$ for each $s \in \mathbf{F}_{2^m}[x, y]$. Thus there are no roots of φ with coordinates in $\mathbf{F}_{2^m}$. In other words, the curve φ has no points over $\mathbf{F}_{2^m}$.

The Hasse–Weil bounds imply, however, that a nonconstant curve of small degree must have points, producing a contradiction if t is small enough. Perhaps one can handle a larger range of t with refined bounds that take account of the special shape of φ; for relevant genus information see, e.g., [3, Theorem 1.3.5].

To summarize: There might exist pairs (ϵ_0, ϵ_1) where ϵ_1 cannot be adjusted to be coprime to h. However, one expects that such pairs do not occur by chance. Furthermore, no such pairs exist if $n < 2^m$, and no such pairs exist if t is small.

A simple (and deterministic) way to handle all the remaining failure cases is to extend the field: any $r \in \mathbf{F}_{2^{2m}} - \mathbf{F}_{2^m}$ has $\epsilon_1 + r\epsilon_0$ coprime to h. The application of $\gcd\{\epsilon_1, h\} = 1$ in Section 7 becomes slower, but still polynomial time, if m is replaced by $2m$.

Numerical example. As in Section 5, define $m = 8$, $n = 2^m = 256$, and $t = 22$; construct $\mathbf{F}_{2^m}$ as $\mathbf{F}_2[\zeta]/(\zeta^8 + \zeta^4 + \zeta^3 + \zeta^2 + 1)$; define $a_1 = \zeta$, $a_2 = \zeta^2$, and so on through $a_{255} = \zeta^{255} = 1$; define $a_{256} = 0$; and choose $g = x^{22} + x^{17} + x^{15} + x^{12} + x^5 + \zeta^{78} \in \mathbf{F}_{2^m}[x]$.

I generated a random element of the Goppa code Γ and added 24 random errors to it, obtaining the word

$$\begin{aligned} w = (&1,0,0,0,1,0,0,1,0,0,0,0,1,1,0,0,1,1,1,0,0,0,1,1,1,1,1,1,0,0,1,0,\\ &0,0,0,1,0,1,1,0,1,0,1,1,0,1,0,0,0,1,0,0,0,0,1,0,0,0,1,0,0,0,1,1,\\ &0,1,1,1,0,0,0,0,1,0,0,0,1,0,0,1,1,1,1,1,0,0,0,1,0,0,0,1,0,1,0,1,0,\\ &0,0,1,0,1,0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,1,1,1,0,1,0,0,0,0,0,1,\\ &0,1,0,0,1,1,1,0,0,1,1,1,0,1,0,1,1,0,1,0,1,0,1,0,0,1,1,0,0,0,1,0,\\ &1,1,0,0,1,0,1,1,1,0,0,0,1,1,0,1,0,1,1,1,1,1,1,1,0,1,1,0,1,0,0,1,0,\\ &0,0,0,1,0,1,0,0,1,1,1,1,1,1,1,0,1,0,0,1,1,0,1,1,1,0,0,1,1,1,1,1,1,\\ &1,1,0,1,0,1,0,1,1,0,0,1,1,1,0,1,1,1,0,0,1,0,0,0,1,0,0,0,1,1,1,0). \end{aligned}$$

Given this word, Patterson's algorithm computes $s = \zeta^{51} + \zeta^{119}x + \zeta^{64}x^2 + \zeta^{230}x^3 + \zeta^9x^4 + \zeta^{30}x^5 + \zeta^{187}x^6 + \zeta^{226}x^7 + \zeta^{55}x^8 + \zeta^{84}x^9 + \zeta^{80}x^{10} + \zeta^{72}x^{11} + \zeta^{71}x^{12} + \zeta^{152}x^{13} + \zeta^{220}x^{14} + \zeta^{221}x^{15} + \zeta^{224}x^{16} + \zeta^{154}x^{17} + \zeta^{166}x^{18} + \zeta^{130}x^{19} + \zeta^{225}x^{20} + \zeta^{11}x^{21}$. Reducing the basis $(s, 1), (g, 0)$ produces a minimum-length nonzero vector (α_0, β_0) and a minimum-length independent vector (α_1, β_1); here $\alpha_0 = \zeta^{52}+\zeta^{27}x+\zeta^{89}x^2+\zeta^{58}x^3+\zeta^{140}x^4+\zeta^{139}x^5+\zeta^{86}x^6+\zeta^{247}x^7+\zeta^{245}x^8+\zeta^{181}x^9+\zeta^{85}x^{10}+\zeta^{37}x^{11}$, $\beta_0 = \zeta^{26} + \zeta^{203}x + \zeta^{175}x^2 + \zeta^{130}x^3 + \zeta^{122}x^4 + \zeta^{168}x^5 + \zeta^{168}x^6 + \zeta^{95}x^7 + \zeta^{154}x^8 + \zeta^{114}x^9 + \zeta^{202}x^{10}$, $\alpha_1 = \zeta^{124} + \zeta^{115}x + \zeta^{194}x^2 + \zeta^{127}x^3 + \zeta^{175}x^4 + \zeta^{84}x^5 + \zeta^{167}x^6 + \zeta^{119}x^7 + \zeta^{55}x^8 + \zeta^{145}x^9 + \zeta^{204}x^{10}$, and $\beta_1 = \zeta^{221} + \zeta^{32}x + \zeta^{113}x^2 + \zeta^{118}x^3 + \zeta^{162}x^4 + \zeta^{93}x^5 + \zeta^{110}x^6 + \zeta^{178}x^7 + \zeta^{67}x^8 + \zeta^{140}x^9 + \zeta^{11}x^{10} + \zeta^{218}x^{11}$.

At this point Patterson's algorithm would hope for $\epsilon_0 = \alpha_0^2 + x\beta_0^2$ to divide h, but there is no such luck; there are no codewords $c \in \Gamma$ with $|w - c| \leq 22$.

The polynomial $\alpha_1^2 + x\beta_1^2$ has roots, as does the polynomial $\alpha_1^2 + x\beta_1^2 + \epsilon_0$, but the polynomial $\epsilon_1 = \alpha_1^2 + x\beta_1^2 + \zeta\epsilon_0$ has no roots; i.e., $\gcd\{\epsilon_1, h\} = 1$. This paper's extension of Patterson's algorithm outputs (ϵ_0, ϵ_1).

Out of curiosity I checked all 256 possibilities for $r \in \mathbf{F}_{2^m}$, and found that a uniform random choice of r has $\gcd\{\alpha_1^2 + x\beta_1^2 + r\epsilon_0, h\} = 1$ with probability $91/256 \approx \exp(-1)$. In retrospect it is not surprising that a few tries sufficed to find a successful value of r.

7 List Decoding via Divisors

Recall that the algorithm from Section 6 finds two polynomials $\epsilon_0, \epsilon_1 \in \mathbf{F}_{2^m}[x]$ such that each desired error polynomial ϵ is a small linear combination of ϵ_0 and ϵ_1. Specifically, if $\deg \epsilon \leq t + u$ and $\deg \epsilon_0 = t_0$ then $\epsilon = q_0^2\epsilon_0 + q_1^2\epsilon_1$ for some polynomials $q_0, q_1 \in \mathbf{F}_{2^m}[x]$ with $\deg q_0 \leq \lfloor (t + u - t_0)/2 \rfloor$ and $\deg q_1 \leq \lfloor (t_0 + u - t - 1)/2 \rfloor$.

A polynomial $\epsilon = q_0^2\epsilon_0 + q_1^2\epsilon_1$ is useful only if its monic part splits into linear factors of the form $x - a_i$; in other words, only if it divides $h = \prod_i (x - a_i)$. Note that q_0, q_1 must be coprime; otherwise ϵ would not be squarefree.

How do we search for divisors of h that are small coprime linear combinations of ϵ_0, ϵ_1? Answer: This is exactly the function-field analogue of the linear-combinations-as-divisors problem solved in Section 3!

To avoid unnecessary dependence on Sections 2 and 3, this section gives a self-contained statement of the list-decoding algorithm. Readers who have studied the algorithm in Section 3 should recognize its similarity to the algorithm in this section.

The list-decoding algorithm. Fix an integer $u \geq 0$. This algorithm will try to correct $t + u$ errors.

Compute ϵ_0, ϵ_1 by the algorithm of Section 6. Define $t_0 = \deg \epsilon_0$; $g_0 = 2\lfloor (u + t - t_0)/2 \rfloor$; $g_1 = 2\lfloor (u + t_0 - t - 1)/2 \rfloor$; and $\theta = g_1 - g_0$.

Adjust ϵ_1, as discussed in Section 6, so that $\gcd\{\epsilon_1, h\} = 1$. Compute a polynomial $\delta \in \mathbf{F}_{2^m}[x]$ such that $\epsilon_1\delta \bmod h = \epsilon_0$.

Fix integers $\ell > k > 0$. Define $L \subset \mathbf{F}_{2^m}(x)[z]$ as the ℓ-dimensional lattice generated by the polynomials

$$1, \frac{x^\theta z + \delta}{h}, \left(\frac{x^\theta z + \delta}{h}\right)^2, \ldots, \left(\frac{x^\theta z + \delta}{h}\right)^k,$$
$$x^\theta z \left(\frac{x^\theta z + \delta}{h}\right)^k, (x^\theta z)^2 \left(\frac{x^\theta z + \delta}{h}\right)^k, \ldots, (x^\theta z)^{\ell-k-1} \left(\frac{x^\theta z + \delta}{h}\right)^k.$$

Use lattice-basis reduction to find a minimal-length nonzero vector $\varphi \in L$. Here the length of $\varphi_0 + \varphi_1 z + \cdots$ is, by definition, $\max\{|\varphi_0|, |\varphi_1|, \ldots\}$.

Use standard polynomial-factorization algorithms to find all of φ's roots in $\mathbf{F}_{2^m}(x)$, and in particular to find roots that have the form $q_0^2/x^\theta q_1^2$ for coprime polynomials $q_0, q_1 \in \mathbf{F}_{2^m}[x]$. For each such root, compute $\epsilon = q_0^2\epsilon_0 + q_1^2\epsilon_1$, and check whether ϵ is a divisor of h; if it is, output the unique $c \in \mathbf{F}_2^n$ such that $\{i : c_i - w_i = 1\} = \{i : \epsilon(a_i) = 0\}$.

What the algorithm accomplishes. Consider any $c \in \Gamma$ with $|w - c| \le t + u$. Define $\epsilon = \prod_{i:w_i \ne c_i}(x - a_i) \in \mathbf{F}_{2^m}[x]$. Then there are polynomials $q_0, q_1 \in \mathbf{F}_{2^m}[x]$, such that $\epsilon = q_0^2\epsilon_0 + q_1^2\epsilon_1$, with $\deg q_0 \le \lfloor (t + u - t_0)/2 \rfloor = g_0/2$ and $\deg q_1 \le \lfloor (t_0 + u - t - 1)/2 \rfloor = g_1/2$; see Section 6.

If $q_0 = 0$ then $\epsilon = q_1^2\epsilon_1$, but ϵ is squarefree, so ϵ/ϵ_1 is a constant, so ϵ_1 divides h, so the algorithm outputs c. Assume from now on that $q_0 \ne 0$.

The fraction $(q_0^2\epsilon_0 + q_1^2\epsilon_1)/h$ is exactly $1/(h/\epsilon)$, so the fraction $(q_1^2 + q_0^2\delta)/h$ is a multiple of $1/(h/\epsilon)$, so the fraction $(q_1^2/q_0^2 + \delta)/h$ is a multiple of $1/(q_0^2 h/\epsilon)$. The value $\varphi(q_1^2/x^\theta q_0^2)$ is a linear combination of

$$1, \frac{q_1^2/q_0^2 + \delta}{h}, \left(\frac{q_1^2/q_0^2 + \delta}{h}\right)^2, \ldots, \left(\frac{q_1^2/q_0^2 + \delta}{h}\right)^k,$$
$$\frac{q_1^2}{q_0^2}\left(\frac{q_1^2/q_0^2 + \delta}{h}\right)^k, \ldots, \left(\frac{q_1^2}{q_0^2}\right)^{\ell-k-1}\left(\frac{q_1^2/q_0^2 + \delta}{h}\right)^k,$$

all of which are multiples of $(1/q_0^2)^{\ell-k-1}(1/(q_0^2 h/\epsilon))^k = 1/(q_0^2)^{\ell-1}(h/\epsilon)^k$. The homogenized value $(q_0^2)^{\ell-1}\varphi(q_1^2/x^\theta q_0^2)$ is therefore a multiple of $1/(h/\epsilon)^k$, which has degree $-k(n - \deg\epsilon)$.

The specified basis elements of L have z-degrees $0, 1, 2, \ldots, k, k+1, k+2, \ldots, \ell-1$ respectively, with leading coefficients

$$1, \frac{x^\theta}{h}, \left(\frac{x^\theta}{h}\right)^2, \ldots, \left(\frac{x^\theta}{h}\right)^k, x^\theta\left(\frac{x^\theta}{h}\right)^k, (x^\theta)^2\left(\frac{x^\theta}{h}\right)^k, \ldots, (x^\theta)^{\ell-k-1}\left(\frac{x^\theta}{h}\right)^k.$$

Thus L is a lattice of dimension ℓ. Furthermore, the product of these leading coefficients is $x^{\theta(\ell-1)\ell/2}/h^{k\ell-k(k+1)/2}$, with degree $\theta(\ell-1)\ell/2+n(k(k+1)/2-k\ell)$. Thus each coefficient of φ has degree at most $\theta(\ell-1)/2+n(k(k+1)/2\ell-k)$.

The degree of q_0^2 is at most g_0, and the degree of q_1^2/x^θ is at most $g_1-\theta=g_0$, so the homogenized value $(q_0^2)^{\ell-1}\varphi(q_1^2/x^\theta q_0^2)=\varphi_0(q_0^2)^{\ell-1}+\varphi_1(q_0^2)^{\ell-2}q_1^2/x^\theta+\cdots+\varphi_{\ell-1}(q_1^2/x^\theta)^{\ell-1}$ has degree at most $\theta(\ell-1)/2+n(k(k+1)/2\ell-k)+(\ell-1)g_0=(g_0+g_1)(\ell-1)/2+n(k(k+1)/2\ell-k)$.

If $\deg\epsilon>(g_0+g_1)(\ell-1)/2k+n(k+1)/2\ell$ then $-k(n-\deg\epsilon)>(g_0+g_1)(\ell-1)/2+n(k(k+1)/2\ell-k)$ so $(q_0^2)^{\ell-1}\varphi(q_1^2/x^\theta q_0^2)$ must be 0. The algorithm finds $q_1^2/x^\theta q_0^2$ as a root of φ, finds (q_0,q_1) since $\gcd\{q_0,q_1\}=1$, finds ϵ, sees that ϵ divides h, and outputs c.

By choosing a moderately large k, and choosing $\ell\approx k\sqrt{n/(g_0+g_1)}$, one can push the degree bound $(g_0+g_1)(\ell-1)/2k+n(k+1)/2\ell$ to approximately $\sqrt{n(g_0+g_1)}\approx\sqrt{2(u-1)n}$. If the degree bound is below $t+u$ then the algorithm will find every codeword at distance $t+u$ from w; if the degree bound is below $t+u-1$ then the algorithm will find every codeword at distance $t+u$ or $t+u-1$ from w; etc. One can cover smaller distances by running the algorithm several times with different choices of u. (See [5, Section 6] for discussion of an analogous loop in the Coppersmith–Howgrave-Graham–Nagaraj algorithm.)

This decoding guarantee breaks down at approximately $n-\sqrt{n(n-2t-2)}$ errors: the degree bound $\sqrt{2(u-1)n}$ grows past $t+u$ as $t+u$ grows past $n-\sqrt{n(n-2t-2)}$.

Numerical example. This example is a continuation of the example in Section 6. Recall that the extension of Patterson's algorithm produced two polynomials $\epsilon_0=\zeta^{74}x^{22}+\cdots$ and $\epsilon_1=\zeta^{181}x^{23}+\cdots$ with $\gcd\{\epsilon_1,h\}=1$. The goal of the algorithm in this section is to find a small linear combination $\epsilon=q_0^2\epsilon_0+q_1^2\epsilon_1$ that divides $h=x^{256}-x$.

Choose $u=2$. Then $t_0=22$, $g_0=2$, $g_1=0$, and $\theta=-2$. The algorithm will search for ϵ of degree $t+u=24$; equivalently, for q_0 of degree $\le g_0/2=1$ and q_0 of degree $\le g_1/2=0$.

Choose $k=8$ and $\ell=87$. Note that $(g_0+g_1)(\ell-1)/2k+n(k+1)/2\ell=2783/116<24$. This example requires a moderately large k, since $t+u=24$ is quite close to $n-\sqrt{n(n-2t-2)}\approx 24.1$.

Divide ϵ_0 by ϵ_1 modulo h to obtain $\delta=\zeta^{200}x^{255}+\zeta^{62}x^{254}+\cdots+\zeta^{85}x+\zeta^{104}$. Define L as the $\mathbf{F}_{2^m}[x]$-submodule of $\mathbf{F}_{2^m}(x)[z]$ generated by

$$1,\frac{z/x^2+\delta}{h},\ldots,\frac{(z/x^2+\delta)^8}{h^8},\left(\frac{z}{x^2}\right)\frac{(z/x^2+\delta)^8}{h^8},\ldots,\left(\frac{z}{x^2}\right)^{78}\frac{(z/x^2+\delta)^8}{h^8}.$$

Then L is an 87-dimensional lattice. The coefficients of $1, z, z^2, \ldots, z^{86}$ in the generators are the columns of the following 87×87 matrix:

$$\begin{matrix}
1 & \delta/h & \delta^2/h^2 & \delta^3/h^3 & \delta^4/h^4 & \delta^5/h^5 & \delta^6/h^6 & \delta^7/h^7 & \delta^8/h^8 & 0 & \cdots & 0 \\
0 & 1/x^2h & 0 & \delta^2/x^2h^3 & 0 & \delta^4/x^2h^5 & 0 & \delta^6/x^2h^7 & 0 & \delta^8/x^2h^8 & \cdots & 0 \\
0 & 0 & 1/x^4h^2 & \delta/x^4h^3 & 0 & 0 & \delta^4/x^4h^6 & \delta^5/x^4h^7 & 0 & 0 & \cdots & 0 \\
0 & 0 & 0 & 1/x^6h^3 & 0 & 0 & 0 & \delta^4/x^6h^7 & 0 & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & 1/x^8h^4 & \delta/x^8h^5 & \delta^2/x^8h^6 & \delta^3/x^8h^7 & 0 & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & 0 & 1/x^{10}h^5 & 0 & \delta^2/x^{10}h^7 & 0 & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1/x^{12}h^6 & \delta/x^{12}h^7 & 0 & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/x^{14}h^7 & 0 & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/x^{16}h^8 & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/x^{18}h^8 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 1/x^{172}h^8
\end{matrix}$$

It is convenient for computation to scale the entire matrix by $x^{172}h^8 = x^{2220} + x^{180}$, to avoid working with fractions. The determinant of the scaled matrix is $x^{7482}h^{36}$, with degree $16698 < 192 \cdot 87$, so lattice-basis reduction is guaranteed to find a nonzero vector $\varphi \in x^{172}h^8L$ where each component has degree < 192.

I reduced the lattice basis and, unsurprisingly, found such a vector, namely $\varphi = \varphi_0 + \varphi_1 z + \cdots + \varphi_{86} z^{86}$ where $\varphi_0 = \zeta^{232}x^{191} + \zeta^{42}x^{190} + \cdots + \zeta^{244}x^{172}$, $\varphi_1 = \zeta^{232}x^{191} + \zeta^{226}x^{190} + \cdots + \zeta^{132}x^{170}$, and so on through $\varphi_{86} = \zeta^{145}x^{191} + \zeta^{10}x^{190} + \cdots + \zeta^{36}x^0$. It turned out that the first 6 successive minima of the lattice all have degree < 192, so there were actually $256^6 - 1$ possibilities for φ.

I then computed the roots of φ in $\mathbf{F}_{2^m}[x]$ and found exactly one root of the desired form: namely, $\varphi(q_1^2/x^\theta q_0^2) = 0$ for $q_1 = \zeta^{153}$ and $q_0 = x - \zeta^{175}$. This calculation was particularly straightforward since the irreducible factorization $\varphi_{86} = \zeta^{145}(x^{170} + \cdots)(x^{13} + \cdots)(x^3 + \cdots)(x^3 + \cdots)(x - \zeta^{175})^2$ had only one square factor. A greatest-common-divisor calculation between leading terms of two independent short vectors would have revealed the same denominator even more easily.

Finally, the sum $\epsilon = q_0^2\epsilon_0 + q_1^2\epsilon_1 = \zeta^{74}x^{24} + \cdots$ has 24 distinct roots, namely ζ^2, $\zeta^6, \zeta^7, \zeta^{15}, \zeta^{23}, \zeta^{38}, \zeta^{46}, \zeta^{59}, \zeta^{71}, \zeta^{73}, \zeta^{86}, \zeta^{88}, \zeta^{131}, \zeta^{138}, \zeta^{142}, \zeta^{150}, \zeta^{153}, \zeta^{159}, \zeta^{163}$, $\zeta^{165}, \zeta^{171}, \zeta^{172}, \zeta^{206}, \zeta^{214}$. Correcting the corresponding positions in w produces the unique $c \in \Gamma$ with $|w - c| \leq 24$.

References

[1] Proceedings of the 32nd Annual ACM Symposium on Theory of Computing. ACM, New York (2000), ISBN 1-58113-184-4, See [8]

[2] Augot, D., Barbier, M., Couvreur, A.: List-decoding of binary Goppa codes up to the binary Johnson bound, `http://arxiv.org/abs/1012.3439`, Citations in this document: §1

[3] Avanzi, R.M.: A study on polynomials in separated variables with low genus factors, Ph.D. thesis, Universität Essen (2001), http://caccioppoli.mac.rub.de/website/papers/phdthesis.pdf, Citations in this document: §6
[4] Bernstein, D.J.: Fast multiplication and its applications. In: [11], pp. 325–384 (2008), http://cr.yp.to/papers.html#multapps, Citations in this document: §1
[5] Bernstein, D.J.: Reducing lattice bases to find small-height values of univariate polynomials. In: [11], pp. 421–446 (2008), http://cr.yp.to/papers.html#smallheight, Citations in this document: §3, §3, §3, §3, §7
[6] Bernstein, D.J., Lange, T., Peters, C.: Attacking and defending the McEliece cryptosystem. In: [10], pp. 31–46 (2008), http://cr.yp.to/papers.html#mceliece, Citations in this document: §1, §4
[7] Bernstein, D.J., Lange, T., Peters, C.: Wild McEliece. In: Biryukov, A., Gong, G., Stinson, D.R. (eds.) SAC 2010. LNCS, vol. 6544, pp. 143–158. Springer, Heidelberg (2011), Citations in this document: §1, §4
[8] Boneh, D.: Finding smooth integers in short intervals using CRT decoding. In: [1], pp. 265–272 (2000); see also newer version [9], Citations in this document: §2
[9] Boneh, D.: Finding smooth integers in short intervals using CRT decoding. Journal of Computer and System Sciences 64, 768–784 (2002), http://crypto.stanford.edu/~dabo/abstracts/CRTdecode.html; see also older version [8], ISSN 0022-0000, MR 1 912 302
[10] Buchmann, J., Ding, J. (eds.): PQCrypto 2008. LNCS, vol. 5299. Springer, Heidelberg (2008), See [6]
[11] Buhler, J.P., Stevenhagen, P. (eds.): Surveys in algorithmic number theory. Mathematical Sciences Research Institute Publications, vol. 44. Cambridge University Press, New York (2008), See [4], [5]
[12] Castagnos, G., Joux, A., Laguillaumie, F., Nguyen, P.Q.: Factoring pq^2 with quadratic forms: nice cryptanalyses. In: [23], pp. 469–486 (2009), Citations in this document: §3
[13] Cohn, H., Heninger, N.: Ideal forms of Coppersmith's theorem and Guruswami-Sudan list decoding (2010), http://arxiv.org/abs/1008.1284, Citations in this document: §3
[14] Coppersmith, D., Howgrave-Graham, N., Nagaraj, S.V.: Divisors in residue classes, constructively (2004), http://eprint.iacr.org/2004/339, see also newer version [15]
[15] Coppersmith, D., Howgrave-Graham, N., Nagaraj, S.V.: Divisors in residue classes, constructively. Mathematics of Computation 77, 531–545 (2008); see also older version [15], Citations in this document: §2
[16] Graham, R.L., Nešetřil, J. (eds.): The mathematics of Paul Erdös. I. Algorithms and Combinatorics, vol. 13. Springer, Berlin (1997), ISBN 3-540-61032-4, MR 97f:00032, See [20]
[17] Guruswami, V., Sudan, M.: Improved decoding of Reed-Solomon and algebraic-geometry codes. IEEE Transactions on Information Theory 45, 1757–1767 (1999), http://theory.lcs.mit.edu/~madhu/bib.html, ISSN 0018-9448, MR 2000j:94033, Citations in this document: §1, §1
[18] Guruswami, V.: List decoding of error-correcting codes, Ph.D. thesis, Massachusetts Institute of Technology (2001), Citations in this document: §1
[19] Howgrave-Graham, N.: Computational mathematics inspired by RSA, Ph.D. thesis (1998), http://cr.yp.to/bib/entries.html#1998/howgrave-graham, Citations in this document: §2

[20] Konyagin, S., Pomerance, C.: On primes recognizable in deterministic polynomial time. In: [16], pp. 176–198 (1997), http://cr.yp.to/bib/entries.html#1997/konyagin, MR 98a:11184, Citations in this document: §2

[21] Lenstra, A.K.: Factoring multivariate polynomials over finite fields. Journal of Computer and System Sciences 30, 235–248 (1985), MR 87a:11124, Citations in this document: §2

[22] Lenstra Jr., H.W.: Divisors in residue classes. Mathematics of Computation 42, 331–340 (1984), http://www.jstor.org/sici?sici=0025-5718(198401)42:165<331:DIRC>2.0.CO;2-6, ISSN 0025-5718, MR 85b:11118, Citations in this document: §2

[23] Matsui, M. (ed.): ASIACRYPT 2009. LNCS, vol. 5912. Springer, Heidelberg (2009), See [12]

[24] McEliece, R.J.: A public-key cryptosystem based on algebraic coding theory. JPL DSN Progress Report, 114–116 (1978), http://ipnpr.jpl.nasa.gov/progress_report2/42-44/44N.PDF, Citations in this document: §1

[25] Mulders, T., Storjohann, A.: On lattice reduction for polynomial matrices. Journal of Symbolic Computation 35, 377–401 (2003), Citations in this document: §2

[26] Niederreiter, H.: Knapsack-type cryptosystems and algebraic coding theory. Problems of Control and Information Theory 15, 159–166 (1986), Citations in this document: §1

[27] Patterson, N.J.: The algebraic decoding of Goppa codes. IEEE Transactions on Information Theory 21, 203–207 (1975), Citations in this document: §1

[28] Sugiyama, Y., Kasahara, M., Hirasawa, S., Namekawa, T.: Further results on Goppa codes and their applications to constructing efficient binary codes. IEEE Transactions on Information Theory 22, 518–526 (1976), Citations in this document: §1, §1

[29] Wu, Y.: New list decoding algorithms for Reed–Solomon and BCH codes. IEEE Transactions On Information Theory 54 (2008), http://arxiv.org/abs/cs/0703105, Citations in this document: §1

Faster 2-Regular Information-Set Decoding

Daniel J. Bernstein[1], Tanja Lange[2], Christiane Peters[2], and Peter Schwabe[3]

[1] Department of Computer Science
University of Illinois at Chicago, Chicago, IL 60607–7045, USA
djb@cr.yp.to
[2] Department of Mathematics and Computer Science
Technische Universiteit Eindhoven, P.O. Box 513, 5600 MB Eindhoven, Netherlands
tanja@hyperelliptic.org, c.p.peters@tue.nl
[3] Institute of Information Science
Academia Sinica, 128 Section 2 Academia Road, Taipei 115-29, Taiwan
peter@cryptojedi.org

Abstract. Fix positive integers B and w. Let C be a linear code over $\mathbf{F}_2$ of length Bw. The 2-regular-decoding problem is to find a nonzero codeword consisting of w length-B blocks, each of which has Hamming weight 0 or 2. This problem appears in attacks on the FSB (fast syndrome-based) hash function and related proposals. This problem differs from the usual information-set-decoding problems in that (1) the target codeword is required to have a very regular structure and (2) the target weight can be rather high, so that there are many possible codewords of that weight.

Augot, Finiasz, and Sendrier, in the paper that introduced FSB, presented a variant of information-set decoding tuned for 2-regular decoding. This paper improves the Augot–Finiasz–Sendrier algorithm in a way that is analogous to Stern's improvement upon basic information-set decoding. The resulting algorithm achieves an exponential speedup over the previous algorithm.

Keywords: Information-set decoding, 2-regular decoding, FSB, binary codes.

1 Introduction

The FSB family of hash functions was submitted to the SHA-3 competition by Augot, Finiasz, Gaborit, Manuel, and Sendrier in 2008 [1]. The submission proposes six specific hash functions: FSB_{160}, FSB_{224}, FSB_{256}, FSB_{384}, FSB_{512}, and a toy version FSB_{48}. The index specifies the size of the output. The hash function consists of a compression function, which is iterated to compress the entire message one block at a time, and a hash function to handle the output of

This work was supported by the National Science Foundation under grant 0716498, by the European Commission under Contract ICT-2007-216499 CACE, and by the European Commission under Contract ICT-2007-216646 ECRYPT II. Permanent ID of this document: c6c2347b09f3864994aefae5f5b6e7be. Date: 2011.03.08.

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 81–98, 2011.
© Springer-Verlag Berlin Heidelberg 2011

the final round of the compression function. The designers chose Whirlpool as the final hash function.

The compression function is what gives this class of functions its name "fast syndrome-based hash functions". The compression function uses a matrix H over $\mathbf{F}_2$ of size $r \times 2^b w$, viewed as having w blocks of size $r \times 2^b$; the parameters here are, e.g., $r = 640$, $b = 14$, $w = 80$ for FSB_{160}. The matrix H is a parity-check matrix for a code of length $2^b w$ and dimension at least $2^b w - r$. A single iteration of the compression function takes as input a bit string of length bw, interprets the bit string as a sequence of w numbers $m_1, m_2, \ldots, m_w$ in $[0, 2^b - 1]$, and computes the sum of the columns indexed by $m_1+1, m_2+1, \ldots, m_w+1$ in blocks $1, 2, \ldots, w$ respectively. The output of the compression function is therefore Hy, the syndrome of the vector $y = ((2^{m_1})_2, (2^{m_2})_2, \ldots, (2^{m_w})_2)$, where $(2^{m_i})_2$ means the 2^b-bit binary representation of 2^{m_i} in little-endian notation. The primary goal of the compression function is to make it difficult for attackers to find a collision, i.e., two distinct inputs that compress to the same output.

Details about how the matrix is constructed and how the message blocks are chained can be found in the design document [1] and in the papers [2], [3], [15], and [14] describing preliminary FSB designs. In [7] we proposed a more efficient family of syndrome-based hash functions called RFSB (for "really fast syndrome-based" hashing); RFSB differs from FSB in the parameter choices and in the way the matrix is constructed. For this paper the matrix-construction details do not matter; for stating the algorithms we consider a general $r \times 2^b w$ matrix, or even more generally an $r \times Bw$ matrix. We use FSB_{160} as an example to illustrate the ideas and the improvements in various algorithms.

Two distinct vectors y and y' having the same syndrome $Hy = Hy'$ do not necessarily correspond to a collision in the compression function, because not every vector can be written in the form $((2^{m_1})_2, (2^{m_2})_2, \ldots, (2^{m_w})_2)$. If the vectors y and y' correspond to colliding messages then they must have Hamming weight exactly 1 in each block. The sum $y + y'$ is then a nonzero 2-regular codeword, where 2-*regular* means that the word has weight 0 or 2 in each block. Note that the concept of 2-regularity for $\mathbf{F}_2^{Bw}$ depends implicitly on the partitioning of Bw positions into w blocks; sometimes we write w-*block* 2-*regularity*.

Conversely, any 2-regular codeword can be written trivially as $y + y'$, where y and y' each have weight exactly 1 in each block. Any nonzero 2-regular codeword therefore immediately reveals a collision in this compression function. The problem of finding a collision is thus equivalent to the problem of 2-*regular decoding*, i.e., the problem of finding a nonzero 2-regular codeword in a code, specifically the code defined by the parity-check matrix H.

There is an extensive literature on algorithms to search for low-weight words in codes. One can use any of these algorithms to search for a codeword of weight $2w$ (or of weight in $\{2, 4, 6, \ldots, 2w\}$), and then hope that the resulting codeword is 2-regular. However, it is better to pay attention to 2-regularity in the design of the low-weight-codeword algorithm. Augot, Finiasz, and Sendrier introduced the first dedicated 2-regular-decoding algorithm in the same 2003 paper [2] that introduced FSB and the 2-regular-decoding problem.

This paper generalizes and improves the Augot–Finiasz–Sendrier algorithm. The new algorithm combines ideas from various improved versions of low-weight information-set decoding, and restructures those ideas to fit the more complicated context of 2-regular codewords. In particular, our attack adapts ideas of Lee–Brickell, Leon, and Stern (see Section 2) to increase the chance of success per iteration at the expense of more effort per iteration. The increase in success chance outweighs the extra effort by an exponential factor.

Section 5 shows the impact of the new algorithm upon FSB_{48}, FSB_{160}, FSB_{256}, and RFSB-509. In each case our algorithm uses far fewer operations than the algorithm of [2]. Note, however, that all of these compression functions are conservatively designed; our algorithm is not fast enough to violate the security claims made by the designers.

All of these algorithms can be generalized to decoding arbitrary syndromes for codes over arbitrary finite fields $\mathbf{F}_q$. The only case that arises in our target application is decoding syndrome 0 over $\mathbf{F}_2$.

Model of computation. Like previous papers on information-set decoding, this paper counts the number of bit operations used for arithmetic, and ignores the cost of memory access. We have made no attempt to minimize the amount of memory used by our new algorithm, and we do not claim that our algorithm is an improvement over previous algorithms in models of computation that penalize memory access.

Other approaches. Information-set decoding is not the only strategy for finding 2-regular codewords. Three other attack strategies have been applied to the FSB collision-finding problem: linearization, generalized birthday attacks, and reducibility. See our survey [7, Section 4] for credits, citations, and corrections.

Information-set decoding, linearization, and generalized birthday attacks are generic techniques that apply to practically all matrices H. Reducibility is a special-purpose technique that relies on a particular structure of H (used in the FSB proposals from [15]) and that is easily combined with the generic techniques when it is applicable. The generic techniques have not been successfully combined with each other, and it is not clear that any one of these techniques is superseded by the others. Linearization seems unbeatable when w/r is not much below $1/2$, but it degrades rapidly in performance as w/r decreases. For small w/r the best technique could be information-set decoding or generalized birthday attacks. The FSB paper [3] says that generalized birthday attacks are a larger threat; the FSB submission [1, Table 4, "best attacks known": "collision search" column] says that information-set decoding is a larger threat; both of the underlying analyses are disputed in [4]. We recommend continuing investigation of all of these approaches.

2 Low-Weight Information-Set Decoding

This section reviews several improvements in low-weight information-set decoding, as background for our improvements in 2-regular information-set decoding.

Types of decoders. We systematically phrase the algorithms here as syndrome-decoding algorithms using parity-check matrices. The goal is to find a low-weight error vector e matching a given syndrome s for a given parity-check matrix H: specifically, to find $e \in \mathbf{F}_2^n$ with $\mathrm{wt}(e) \leq t$ such that $He = s$, given $s \in \mathbf{F}_2^r$ and $H \in \mathbf{F}_2^{r \times n}$. We abbreviate $n - r$ as k.

These algorithms can be, and in the literature often are, translated into word-decoding algorithms using generator matrices. The distinction between syndrome decoding and word decoding is minor: an application that wants word decoding can begin with a word $v \in \mathbf{F}_2^n$, compute the syndrome $s = Hv$, apply a syndrome-decoding algorithm to find e, and finally compute $v - e$, a codeword whose distance from v is at most t. The distinction between parity-check matrices and generator matrices is more important, and can have a noticeable effect on the efficiency of the algorithms, although we are not aware of any systematic study of this effect. It is typical in code-based cryptography for k to be larger than $n/2$, so parity-check matrices are smaller than generator matrices; this is particularly obvious for the parameters n and k appearing in FSB.

Plain information-set decoding. Information-set decoding was first suggested by Prange in [22] and was later used by McEliece [21] to estimate the security of code-based cryptography.

One iteration of plain information-set decoding works as follows. Select a random set of r columns of the $r \times n$ parity check matrix H of the code. Permute columns to move these to the right-hand side of the matrix, producing a matrix H'. Compute the inverse U of the rightmost $r \times r$ submatrix of H'; if the submatrix is not invertible — i.e., if the k non-selected columns are not an information set — then the iteration fails. If $\mathrm{wt}(Us) \leq t$ then the iteration has successfully found a low-weight error vector $e' = (0 \ldots 0 | Us)$ matching Us for $UH' = (\cdots | I_r)$ and therefore matching s for H'; reversing the column permutation on the positions of e' produces a low-weight error vector e matching s for H. Otherwise the iteration fails. The method succeeds if there are no errors located in the information set.

As mentioned in the introduction, our main concern in this paper is the case $s = 0$. Prange's algorithm is not interesting for $s = 0$: its only possible output is 0. Weight-t codewords can be viewed as weight-t error vectors (relative to codeword 0) but will not be found by this algorithm (except in the degenerate case $t = 0$). The improved algorithms discussed below do not have this limitation, and remain interesting in the case $s = 0$: they allow sums of columns of I_r to be cancelled by sums of other columns of UH', so they can find nonzero error vectors having syndrome 0. A different way to handle syndrome 0 is to extend Prange's algorithm to scan the entire kernel of the $r \times r$ submatrix of H', allowing the submatrix to be non-invertible; this is the starting point for the algorithm of [2] discussed in the next section.

The standard improvements. Lee and Brickell in [18] improved Prange's method by choosing a small parameter $p \leq t$ and allowing p errors in the information set (together with $\leq t - p$ errors in the selected columns). This means

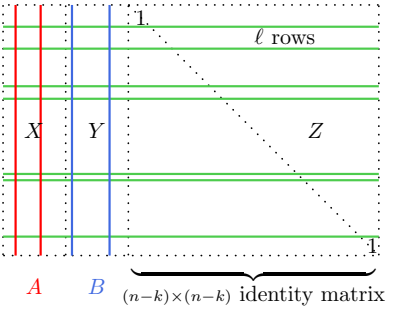


Fig. 2.1. One check in Stern's algorithm

checking $\binom{k}{p}$ combinations of columns but amortizes the costs of Gaussian elimination across those combinations.

Leon in [19], independently of Lee and Brickell, suggested p errors in the information set together with ℓ-row early aborts. Instead of checking the weight of the sum of each set of p columns, this algorithm checks only the sets that add up to 0 on a subset Z of the rows, where Z has size ℓ. The effort for each of the $\binom{k}{p}$ combinations is reduced from an r-row sum to an ℓ-row sum (plus an $(r-\ell)$-row sum with probability about $1/2^{\ell}$), at the cost of missing error vectors that have errors in the ℓ columns corresponding to Z.

The next year, in [24], Stern suggested the same improvements together with a collision speedup. The information set is partitioned into two sets X and Y. The Lee–Brickell parameter p is required to be even and is split as $(p/2)+(p/2)$. The algorithm searches for error vectors that have $p/2$ errors among the positions in X and $p/2$ errors among the positions in Y; all $(p/2)$-column subsets A of X are tested for matches with all $(p/2)$-column subsets B of Y. The algorithm follows Leon's by checking only the pairs (A, B) that add up to 0 on a subset Z of the rows. The collision speedup is as follows: instead of trying all pairs (A, B), the algorithm computes one list of Z-sums of $(p/2)$-column subsets A, and a second list of Z-sums of $(p/2)$-column subsets B, and then efficiently finds collisions between the two lists.

Figure 2.1 displays one checking step in Stern's algorithm for $p = 4$ and $\ell = 7$. The two leftmost solid columns (red on color displays) form the set A; the two rightmost solid columns (blue) form the set B. The pair (A, B) is considered only if these columns match on the ℓ positions indicated by solid rows (green), i.e., sum up to 0 on each of those positions. If so, the sum is computed on the full length r. If the sum has weight $t-4$ then the algorithm has found a word of weight t. This word has nonzero entries in the 4 columns indexed by A, B and the positions where the $t-4$ errors occur.

Further improvements. Many papers have proposed improvements to Stern's algorithm, for example in how the matrices H' and UH' are computed; in how the choices of columns in X and Y are handled; and in how the full test is done

once a choice was successful on the ℓ positions. The most recent papers are [5], [16], and [6]; see those papers for surveys of previous work.

3 The Augot–Finiasz–Sendrier Algorithm for 2-Regular Decoding

This section discusses the Augot–Finiasz–Sendrier algorithm [2, Section 4.2] for 2-regular decoding. The algorithm inputs are positive integers r, B, w and a parity-check matrix $H \in \mathbf{F}_2^{r \times n}$, where $n = Bw$. If the algorithm terminates then it outputs a nonzero w-block 2-regular codeword; recall that this means a nonzero codeword v such that, for each $i \in \{1, 2, \ldots, w\}$, the ith B-bit block of v has Hamming weight 0 or 2.

The algorithm can be generalized to decoding arbitrary syndromes, but we focus on syndrome 0 as discussed in the introduction. We refer to the positions of nonzero entries in the target codeword as error positions, viewing the target codeword as an error vector relative to codeword 0.

Review of the algorithm. Each iteration of this algorithm selects a set of r out of the n positions of columns from H, performs $r \times r$ Gaussian elimination to compute the kernel of those r columns, and checks each nonzero element of the kernel for 2-regularity. The selection is split evenly among w_0 blocks of H, where $w_0 \in \{1, 2, 3, \ldots, w\}$ is an algorithm parameter; assume for the moment that w_0 divides r and that $r/w_0 \leq B$.

Out of all 2^r vectors supported in these r positions, only $\sum_{1 \leq i \leq w_0} \binom{r}{2i}$ have weight in $\{2, 4, 6, \ldots, 2w_0\}$, and only $\left(\binom{r/w_0}{2} + 1\right)^{w_0} - 1$ are nonzero 2-regular vectors. Each of these nonzero 2-regular vectors has, under suitable randomness assumptions on H, probability $1/2^r$ of being a codeword, so the expected number of codewords found by one iteration of the algorithm is $\left(\left(\binom{r/w_0}{2} + 1\right)^{w_0} - 1\right)/2^r$.

Augot, Finiasz, and Sendrier conclude that the success probability of an iteration is $\left(\binom{r/w_0}{2} + 1\right)^{w_0}/2^r$. Here they are ignoring the -1 above, and ignoring the difference between the success probability and the expected number of codewords (i.e., ignoring the possibility of an iteration finding two codewords at once), but these are minor effects.

The expected number of kernel elements is (again under suitable randomness assumptions on H, which we now stop mentioning explicitly) a constant; a large kernel occurs with only small probability. The bottleneck in the iteration is Gaussian elimination, using $O(r^3)$ bit operations.

Non-divisibility. Augot, Finiasz, and Sendrier say that for $w < \alpha r$ it is best to take $w_0 = w$; here $\alpha \approx 0.24231$ is chosen to maximize $\left(\binom{1/\alpha}{2} + 1\right)^{\alpha}$. Many of the published FSB parameters (r, B, w) have w dividing r and $w < \alpha r$; in all of these cases, w_0 will also divide r.

However, Augot, Finiasz, and Sendrier also consider many parameters with $w > \alpha r$, and say that in this case it is best to take $w_0 = \alpha r$. Presumably this

means choosing w_0 as an integer very close to αr, but for almost all values of r this means that w_0 cannot divide r. This non-divisibility causes various problems that are not discussed in [2] and that invalidate some of the algorithm analysis in [2], as we now show.

The obvious way to interpret the algorithm to cover this case is to take some blocks with $\lfloor r/w_0 \rfloor$ selected columns, and some with $\lceil r/w_0 \rceil$, for a total of r columns. Write $f = \lfloor r/w_0 \rfloor$, $b = r - w_0 f = r \bmod w_0$, and $a = w_0 - b$; then one can take a blocks each with f columns and b blocks each with $f + 1$ columns, for a total of r columns in w_0 blocks. For example, if $w_0 = 0.24r$, then one can take $5w_0 - r = 0.20r$ blocks with 4 columns and $r - 4w_0 = 0.04r$ blocks with 5 columns.

The number of 2-regular words supported in these r columns is exactly $\left(\binom{f}{2} + 1\right)^a \left(\binom{f+1}{2} + 1\right)^b$. Here we are counting, for each of the a blocks, the number of ways to choose 0 or 2 out of f columns; and, for each of the b blocks, the number of ways to choose 0 or 2 out of $f+1$ columns. The expected number of nonzero 2-regular codewords found by one iteration is thus

$$\left(\left(\binom{f}{2} + 1\right)^a \left(\binom{f+1}{2} + 1\right)^b - 1\right) \Big/ 2^r.$$

If $r/5 < w_0 \le r/4$ then this number is $(7^a 11^b - 1)/2^r = (7^{5w_0 - r} 11^{r - 4w_0} - 1)/2^r \approx ((11/14)(7^5/11^4)^{w_0/r})^r$; e.g., approximately $2^{-0.300r}$ for $w_0 = \alpha r$.

The analysis in [2] incorrectly applies the formula $\left(\binom{r/w_0}{2} + 1\right)^{w_0} / 2^r$ without regard to the question of whether r/w_0 is an integer, and in particular for the case $w_0 = \alpha r$. This overstates the success probability by a small but exponential factor, for example claiming success probability $2^{-0.298r}$ for $w_0 = \alpha r$. The discrepancy is larger for ratios r/w_0 that are farther from integers; see Figure 3.1. Many of the curves plotted in [2] and [3, Section 4.4] need to be adjusted accordingly. This analysis also shows that the correct cutoff for w_0 is $0.25r$, not αr. We are not aware of any sensible algorithm modification that would rescue the analysis in [2].

Comparison to low-weight decoding. We emphasize that finding a nonzero 2-regular codeword is much harder than merely finding a word of weight $2w$. Each nonzero 2-regular codeword has weight in $\{2, 4, 6, \ldots, 2w\}$, but the opposite is very far from being true: most words of this weight will not have the right distribution of nonzero entries.

For example, consider FSB_{160}, with $r = 640$, $B = 2^{14}$, and $w = 80$. The parity check matrix is a $640 \times n$ matrix where $n = 2^{14} \cdot 80 = 1310720$. The Augot–Finiasz–Sendrier algorithm picks 640 columns in a regular pattern by taking 8 columns of each of the 80 blocks; by linear algebra identifies all codewords supported in those columns; and checks 2-regularity of each nonzero codeword. The number of nonzero 2-regular vectors supported in those columns is $(\binom{8}{2} + 1)^{80} - 1 \approx 2^{388.64}$, so the number of nonzero 2-regular codewords supported in these columns is expected to be approximately $2^{388.64}/2^{640} = 2^{-251.36}$.

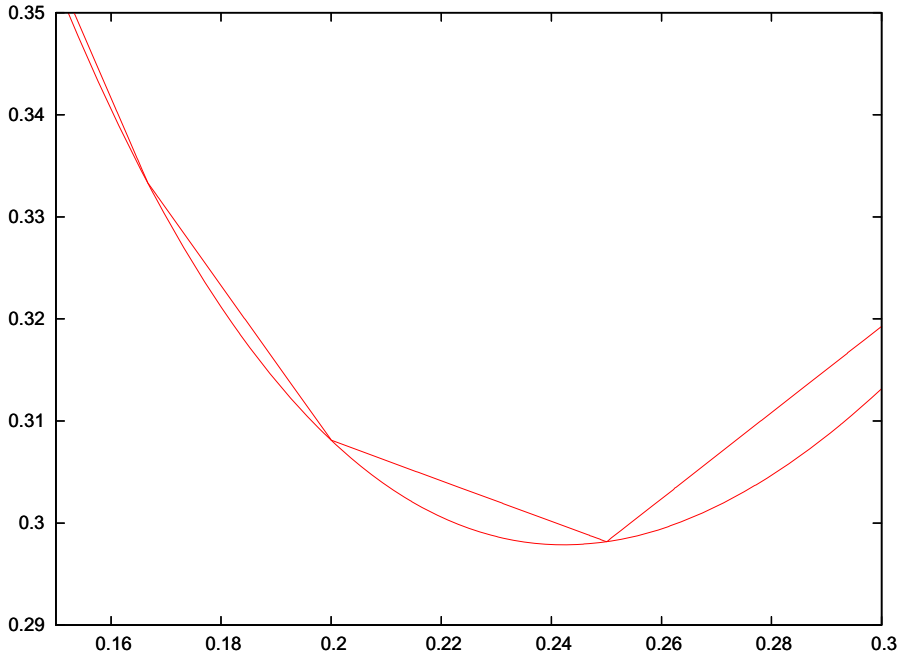


Fig. 3.1. Vertical axis, bottom curve: $y = 1 - x \log_2 \left(\binom{1/x}{2} + 1 \right)$; the Augot–Finiasz–Sendrier algorithm was claimed to use asymptotically 2^{yr} iterations if $x = w_0/r$. Vertical axis, top segments: $y = 1 - (x(f+1) - 1) \log_2 \left(\binom{f}{2} + 1 \right) - (1 - xf) \log_2 \left(\binom{f+1}{2} + 1 \right)$ where $f = \lfloor 1/x \rfloor$; the algorithm actually uses asymptotically 2^{yr} iterations if $x = w_0/r$.

For comparison, the number of weight-160 vectors supported in the same columns is approximately $\binom{640}{160} \approx 2^{514.44}$, so the number of weight-160 codewords supported in these columns is expected to be approximately $2^{514.44}/2^{640} = 2^{-125.56}$. The probability is slightly larger if weights $158, 156, \ldots$ are also allowed. This change in success criteria does not noticeably slow down the iteration, and it drastically reduces the number of iterations required for success.

This difference is even more extreme for $w \approx r/4$. Finding a w-block 2-regular codeword then takes an exponential number of iterations, approximately

$$2^r \Big/ \left(\binom{4}{2} + 1 \right)^{r/4} = (16/7)^{r/4},$$

while finding a weight-$2w$ codeword takes a polynomial number of iterations, approximately

$$2^r \Big/ \binom{r}{r/2} \approx 2^r \left(\sqrt{2\pi r/2} \left(\frac{r/2}{e} \right)^{r/2} \right)^2 \Big/ \left(\sqrt{2\pi r} \left(\frac{r}{e} \right)^r \right) = \sqrt{\pi r/2}.$$

In both cases each iteration takes polynomial time.

4 A New Algorithm for 2-Regular Decoding

This section presents a new algorithm for 2-regular decoding. This algorithm combines the standard improvements in low-weight decoding (see Section 2) with the Augot–Finiasz–Sendrier algorithm described in the previous section.

Impact of the improvements. One might guess that these improvements actually have very little effect, if any at all, upon the complexity of 2-regular decoding. There are several arguments for this guess:

- In the usual context of low-weight decoding, the standard improvements are often viewed as rather small. For example, Augot, Finiasz, and Sendrier in [2, Section 4.1] say that [10], [18], and [24] merely "reduce the degree of the polynomial part" of the complexity of information-set decoding.
- Plain information-set decoding and the Augot–Finiasz–Sendrier algorithm apply linear algebra to an $r \times r$ matrix. The improved algorithms by Lee–Brickell, Leon, and Stern start by inverting an $r \times r$ matrix but then have to multiply the inverse U by the remaining $r \times k$ submatrix of the parity-check matrix H. This extra multiplication has comparable cost to the inversion if k and r are on the same scale, as they usually are in applications of low-weight decoding; but k is usually much larger than r in applications of 2-regular decoding. For example, recall that FSB_{160} has $r = 640$ and $k = 1310080$.
- Speedups in low-weight decoding algorithms are usually aimed at the case of small weight (at most the Gilbert–Varshamov bound), where one expects to find at most one codeword. Normally 2-regular decoding is applied for much larger weights, where many solutions exist. There is no reason to think that speedups in one context should be effective for the other.

But there are also counterarguments. One can show that Stern's speedup is superpolynomial when parameters are properly optimized, and that the cost of linear algebra inside Stern's algorithm is asymptotically negligible. See [8]. For the same reasons, the cost of multiplying U by H is also negligible.

To firmly settle the arguments we show that our new algorithm for 2-regular decoding is faster than the old algorithm by an exponential factor. For example, for $w/r = 0.1435$, the number of bit operations in the new algorithm is $2^{0.2825r}$ times a polynomial factor (provided that $B \geq 2^8$), while the number of bit operations in the old algorithm is $2^{0.3621r}$ times a polynomial factor.

We also evaluate the polynomial factors in the operation count for the new algorithm. The next section considers various specific hash functions, showing in each case the concrete speedup from the Augot–Finiasz–Sendrier algorithm to our algorithm.

The new algorithm. Each iteration of this algorithm works as follows. Select a set of r out of the n positions of columns from H. Split the selection almost evenly (see below) among w_0 blocks of H, where $w_0 \in \{1, 2, 3, \ldots, w\}$ is an algorithm parameter with $r \leq Bw_0$.

The new algorithm will do more work than the old algorithm in this iteration: it will search for 2-regular codewords that have exactly 2ш errors in the $n-r$ non-selected positions. Here $ш \in \{1, \ldots, \lfloor w_0/2 \rfloor\}$ is another algorithm parameter. Note that the presence of 2ш errors will exclude the possibility of uselessly finding codeword 0.

Use Gaussian elimination to see whether the r selected vectors are linearly independent. This occurs with probability approximately 29%; see [11]. If the vectors are dependent, start the iteration over with a new selection of r positions; even with this restarting, Gaussian elimination is not a bottleneck for large ш. An alternative is to consider each kernel element, but for simplicity and speed we enforce linear independence.

The set of non-selected positions is now an information set for H, and the set of selected positions is the corresponding redundancy set. Gaussian elimination has also revealed a linear transformation that converts the selected r vectors into an $r \times r$ identity matrix; apply the same transformation to all of H. This uses quite a few operations, but it is not a bottleneck for large ш.

Assume for simplicity that w_0 is even. Partition the w_0 selected blocks of H into $w_0/2$ "left" blocks and $w_0/2$ "right" blocks; the codewords found by the algorithm will have exactly ш information-set errors spread among exactly ш left blocks, and exactly ш information-set errors spread among exactly ш right blocks. Also choose a set S of ℓ out of the r row indices, where ℓ is another algorithm parameter. This set S corresponds, via the $r \times r$ identity matrix, to ℓ elements of the redundancy set; the codewords found by the algorithm will have 0 errors in those positions, as in Stern's algorithm.

Build a list L as follows. Choose ш left blocks, choose one information-set position in each block, and add the S-indexed bits of those ш vectors, obtaining an ℓ-bit vector. Store the resulting ℓ-bit vector along with the chosen positions as the first entry in L. Repeat until L has N elements, where N is another algorithm parameter. Similarly build a list R of N elements, using the right blocks.

Find all ℓ-bit collisions between L and R. For each collision, compute the sum of the non-S-indexed bits of those 2ш vectors. If the sum has weight $2i - 2ш$ for some $i \in \{2ш, 2ш + 1, \ldots, w_0\}$ then it can be written trivially as a sum of $2i-2ш$ vectors from the redundancy set. The positions of those vectors, together with the positions from L and R, form a codeword of weight $2i$. Output the codeword if it is 2-regular.

Algorithm analysis. Write ℓ as $\ell_1 w_0 + \ell_0$ with $0 \le \ell_0 < w_0$. Our algorithm analysis assumes that $r-\ell$ is a multiple of w_0, say fw_0. In this case the algorithm can, and we assume that it does, select columns as follows: ℓ_0 blocks each contain exactly $f + \ell_1 + 1$ elements of the redundancy set, including $\ell_1 + 1$ elements corresponding to elements of S; $w_0 - \ell_0$ further blocks each contain exactly $f + \ell_1$ elements of the redundancy set, including ℓ_1 elements corresponding to elements of S. Note that each of these w_0 blocks contains exactly f non-S elements of the redundancy set.

This appears to be the most nicely balanced case of the algorithm. However, we do not assert that it is always optimal. The algorithm does not require w_0 to divide $r - \ell$; if w_0 does not divide $r - \ell$ then one can choose sizes so that the total matches and some sets have one extra element each, as in the previous section. We exclude this possibility solely to simplify the analysis.

An element of L and an element of R together specify a pattern of 2ш errors in 2ш blocks in the information set. For each of those blocks there are exactly f ways to choose a non-S element of the redundancy set within the block. For each of the $w_0 - 2\text{ш}$ remaining blocks there are exactly $1 + \binom{f}{2}$ ways to choose zero or two non-S elements of the redundancy set.

Putting together all of these choices produces a nonzero 2-regular error pattern. Each of the 2ш initially specified blocks contains exactly one error in the information set and exactly one non-S error in the redundancy set. Each of the $w_0 - 2\text{ш}$ remaining blocks contains exactly zero or exactly two non-S errors in the redundancy set. If this error pattern is a codeword then it will be found by the algorithm.

The expected number of codewords obtainable in this way is

$$\delta = \frac{N^2}{2^r} f^{2\text{ш}} \left(1 + \binom{f}{2}\right)^{w_0 - 2\text{ш}}$$

under suitable randomness assumptions. The factor N^2 counts the number of pairs of elements of L and elements of R, and the factor $1/2^r$ is the chance of an error pattern being a codeword. The success probability of an iteration is approximately $1 - \exp(-\delta)$.

The cost of an iteration is the cost of linear algebra, plus $2N\ell$ additions for the elements of L and R (assuming reuse of additions as in [5, Section 4]), plus approximately $(N^2/2^\ell)2\text{ш}(r - \ell)$ additions to handle the partial collisions. We could use early aborts as in [5] to reduce the factor $r - \ell$ here, but for simplicity we avoid doing so.

Parameter selection. For various choices of (r, B, w) we performed computer searches to identify good algorithm parameters $(w_0, \text{ш}, N, \ell)$. See Section 5 for examples. The search space is small enough that no general recommendations for parameter choices are needed, but we nevertheless make a few comments regarding the optimal parameters.

There are obvious benefits to increasing N in this algorithm: δ grows quadratically with N, while the cost of the iteration grows only linearly with N (assuming $N < 2^\ell$; see below). But there are two hard limits upon this benefit. First, N cannot exceed the number of choices of entries in L; this number is between $\binom{w_0/2}{\text{ш}}(B-1-r/w_0)^{\text{ш}}$ and $\binom{w_0/2}{\text{ш}}(B+1-r/w_0)^{\text{ш}}$. Second, the quadratic growth of δ with N stops producing a quadratic growth in the success probability of the iteration as δ approaches and passes 1, i.e., as N^2 approaches and passes $2^r f^{-2\text{ш}} \left(1 + \binom{f}{2}\right)^{2\text{ш} - w_0}$. Our computations suggest that, in general, the best operation counts for this algorithm balance the first and second limits: most of the possibilities for L and R are used, and a single iteration has a high chance of success.

If N is allowed to grow beyond 2^ℓ then the cost of the iteration begins to grow quadratically. One can compensate for this by increasing ℓ. In general it seems best to choose ℓ close to $\log_2 N$, as in previous information-set-decoding algorithms, so that the cost of building lists L and R is balanced with the costs of doing the full-length checks. Increasing ℓ has the disadvantage of directly reducing δ, but this appears to be outweighed by the advantage of keeping $N^2/2^\ell$ under control. Beware that increasing ℓ also has a disadvantage not visible in the bit-operation cost model: efficient collision detection needs about 2^ℓ bits of memory.

Asymptotics. Fix a positive integer B and a positive real number W. Assume that $w/r \to W$ as $r \to \infty$. The following analysis optimizes choices of positive real numbers $W_0, \text{Ш}, T$ for the following goals: if the algorithm parameters $w_0, \text{ш}, N, \ell$ are chosen so that $w_0/r \to W_0$, $\text{ш}/r \to \text{Ш}$, $(\log_2 N)/r \to T$, and $\ell/r \to T$, then the algorithm run time also satisfies $(\log_2 \text{time})/r \to T$; furthermore, T is as small as possible.

We impose several constraints upon the choices of $W_0, \text{Ш}, T$:

- The ratio $f = (1-T)/W_0$ is a positive integer. This allows ℓ and w_0 to be chosen so that $(r-\ell)/w_0 = f$ once r is sufficiently large. We suspect that our algorithm achieves smaller exponents without this constraint, but as noted above our algorithm analysis requires w_0 to divide $r - \ell$.
- $W_0 \le W$. This allows w_0 to be chosen in $\{2, 4, 6, \ldots, 2\lfloor w/2 \rfloor\}$.
- $\text{Ш} \le W_0/2$. This allows ш to be chosen in $\{1, 2, 3, \ldots, \lfloor w_0/2 \rfloor\}$.
- $2T - 1 + 2\text{Ш} \log_2 f + (W_0 - 2\text{Ш}) \log_2(1 + f(f-1)/2) = 0$; in other words, $(\log_2 \delta)/r \to 0$. This ensures that the algorithm succeeds within $2^{o(r)}$ iterations. We do not have a proof that this constraint is always optimal, but we impose it here to simplify the asymptotic analysis.
- $T \le (W_0/2) \log_2(W_0/2) - \text{Ш} \log_2 \text{Ш} - (W_0/2 - \text{Ш}) \log_2(W_0/2 - \text{Ш}) + \text{Ш} \log_2(B - 1 - 1/W_0)$. This ensures that N can be chosen below $\binom{w_0/2}{\text{ш}}(B - 1 - r/w_0)^{\text{ш}}$.

Under these constraints, the cost of an iteration is within a polynomial factor of $2^{(T+o(1))r}$, so the total number of bit operations used by the algorithm is also within a polynomial factor of $2^{(T+o(1))r}$.

We view this constrained optimization problem as a series of separate problems: one with $f = 1$, one with $f = 2$, one with $f = 3$, etc. For any particular $f < 1/W_0$, substituting $T = 1 - fW_0$ into $2T - 1 + 2\text{Ш} \log_2 f + (W_0 - 2\text{Ш}) \log_2(1 + f(f-1)/2) = 0$ produces an equation for Ш in terms of W_0, namely

$$\text{Ш} = \frac{1 - (2f - \log_2(1 + f(f-1)/2))W_0}{2\log_2(1 + f(f-1)/2) - 2\log_2 f}.$$

If this does not satisfy $0 < \text{Ш} \le W_0/2$ then f and W_0 are incompatible. The final inequality $T \le \cdots$ then puts a lower bound on B. To summarize, each choice of W_0 has a finite list of possibilities for f, with each possibility immediately dictating Ш, T, and a lower bound on B.

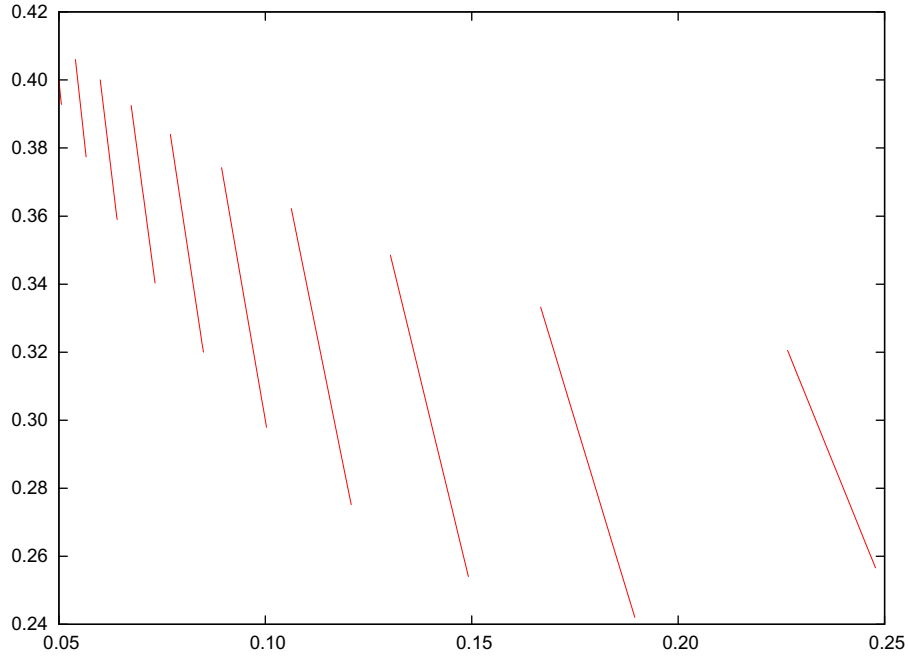


Fig. 4.1. T as a function of W_0. See text for description.

Figure 4.1 shows values of T that can be achieved in this way; Figure 4.2 shows the corresponding lower bounds on $\log_2 B$, and Figure 4.3 shows the corresponding Ш. The horizontal axis in each case is W_0. The values of f are, from right to left, first 3, then 4, etc. The figures omit values of W_0 that require $B > 2^{20}$.

For example, $W_0 = 0.2453$ and $f = 3$ produce $T = 0.2641$, Ш ≈ 0.022649, and the lower bound $B \geq 2^8$. As another example, $W_0 = 0.1435$ and $f = 5$ produce $T = 0.2825$, Ш ≈ 0.027001, and the lower bound $B \geq 2^8$. The smallest value of T in Figure 4.1 is $T = 0.2420$, achieved for $W_0 = 0.1895$, $f = 4$, Ш ≈ 0.009905, and $B \geq 2^{19.81}$.

Given W we scan through $W_0 \leq W$ to minimize T. For example, any $W \geq 0.1435$ can use $T = 0.2825$ by taking $W_0 = 0.1435$, if $B \geq 2^8$. Figure 4.4 plots the resulting T as a function of W, and for comparison plots the exponent of the Augot–Finiasz–Sendrier algorithm.

Further improvements. Finiasz and Sendrier [16] improved Stern's algorithm by avoiding the static split into left and right in the information set. We adapt this approach to the situation of finding 2-regular words as follows. Build one list L by repeatedly picking ш blocks and then for one column per block computing the sum on the ℓ positions specified by S. This increases the maximal value of N to approximately $\binom{w_0}{ш}(B - r/w_0)^{ш}$. Then search for ℓ-bit collisions within L. If a

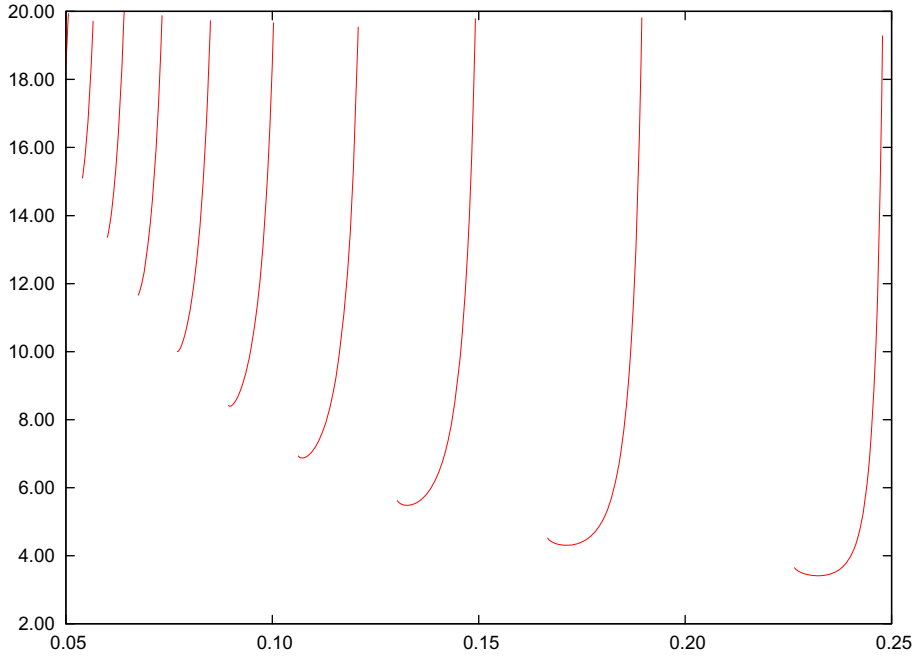


Fig. 4.2. Minimum $\log_2 B$ as a function of W_0. See text for description.

collision on the ℓ positions happens to involve j positions shared between the two entries from L the algorithm can still produce a 2-regular vector. The condition changes to requiring that each of the other $2(ш - j)$ initially specified blocks contains exactly one error in the information set and exactly one non-S error in the redundancy set. Each of the $i - 2ш$ remaining blocks contains exactly two non-S errors in the redundancy set. If this error pattern is a codeword then it will be found by the algorithm. Various computer searches did not find parameters leading to smaller operation counts than before even though this approach can produce a larger N.

The literature for low-weight information-set decoding also contains many improvements to the cost of linear algebra. Adapting these improvements to the 2-regular context might be helpful for small $ш$, but linear algebra becomes negligible as $ш$ grows, so we have not incorporated these improvements into our algorithm. We have also not tried to adapt ball-collision decoding [6] to the 2-regular context.

5 Applications to Hash Functions

This section gives examples of the cost of finding collisions in several different syndrome-based compression functions: the FSB_{48}, FSB_{160}, and FSB_{256} proposals from [1], and the RFSB-509 proposal from [7]. We repeat a caveat from

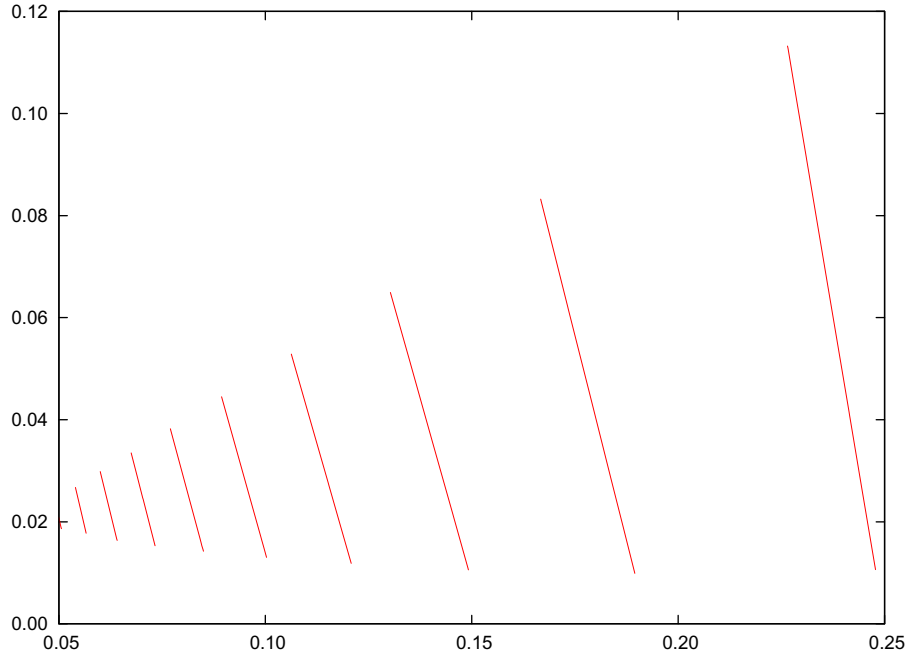


Fig. 4.3. Ш as a function of W_0. See text for description.

Section 1: like previous papers, this paper merely counts the number of bit operations used for arithmetic; we do not claim that our algorithm is an improvement over previous algorithms in models of computation that penalize memory access.

FSB_{48} was designed for $\geq 2^{24}$ collision resistance. It has $r = 192$, $B = 2^{14}$, and $w = 24$. The Augot–Finiasz–Sendrier algorithm needs $2^{75.41}$ iterations, with each iteration using quite a few bit operations. Our algorithm uses just $2^{66.31}$ bit operations with $w_0 = 24$, $ш = 3$, $N \approx 2^{49.78}$, and $\ell = 48$.

FSB_{160} was designed for $\geq 2^{80}$ collision resistance. It has $r = 640$, $B = 2^{14}$, and $w = 80$. The Augot–Finiasz–Sendrier algorithm needs $2^{251.36}$ iterations. Our algorithm uses just $2^{196.70}$ bit operations with $w_0 = 76$, $ш = 11$, $N \approx 2^{182.09}$, and $\ell = 184$.

Augot, Finiasz, Gaborit, Manuel, and Sendrier in [1, Table 4, "best attacks known"] claim a "complexity" of "$2^{100.3}$" for "ISD collision search" against FSB_{160}. In fact, no attacks are known that find FSB_{160} collisions (equivalently, 2-regular codewords) at this speed. The text in [1, Section 2.2.2] makes clear that [1] is merely estimating the cost of finding a weight-160 codeword, not the cost of finding an 80-block 2-regular codeword.

FSB_{256} was designed for $\geq 2^{128}$ collision resistance. It has $r = 1024$, $B = 2^{14}$, and $w = 128$. The Augot–Finiasz–Sendrier algorithm needs $2^{402.18}$ iterations. Our algorithm uses just $2^{307.56}$ bit operations with $w_0 = 122$, $ш = 17$, $N \approx 2^{286.92}$, and $\ell = 292$.

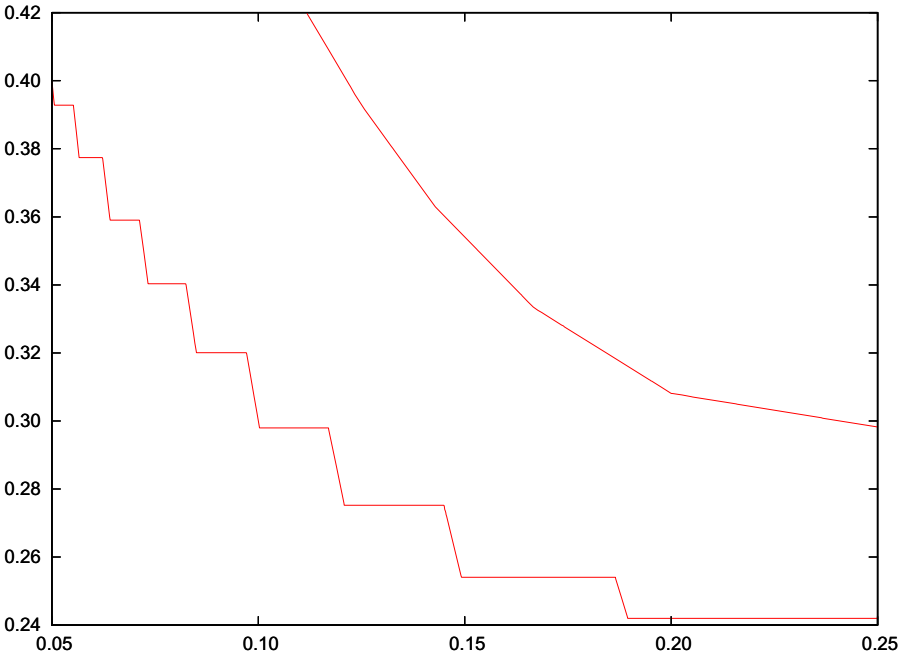


Fig. 4.4. Bottom segments: Asymptotic exponent T for this paper's algorithm (with the restrictions that w_0 divides $r-\ell$ and that $\delta \approx 1$), as a function of W. Top segments: Asymptotic exponent for the Augot–Finiasz–Sendrier algorithm.

RFSB-509 was also designed for $\geq 2^{128}$ collision resistance, but for speed it uses tighter parameters (and a different matrix structure). It has $r = 509$, $B = 2^8$, and $w = 112$. The Augot–Finiasz–Sendrier algorithm needs $2^{154.80}$ iterations, and the Augot–Finiasz–Sendrier analysis would have claimed $2^{152.99}$ iterations. Our algorithm uses just $2^{144.90}$ bit operations with $w_0 = 94$, $\text{III} = 12$, $N \approx 2^{130.75}$, and $\ell = 133$.

References

1. Augot, D., Finiasz, M., Gaborit, P., Manuel, S., Sendrier, N.: SHA-3 proposal: FSB (2008), `http://www-rocq.inria.fr/secret/CBCrypto/fsbdoc.pdf`, Citations in this document: §1, §1, §1, §5, §5, §5, §5
2. Augot, D., Finiasz, M., Sendrier, N.: A fast provably secure cryptographic hash function (2003), `http://eprint.iacr.org/2003/230`, Citations in this document: §1, §1, §1, §2, §3, §3, §3, §3, §3, §3, §4
3. Augot, D., Finiasz, M., Sendrier, N.: A family of fast syndrome based cryptographic hash functions. In: Mycrypt 2005 [13], pp. 64–83 (2005), Citations in this document: §1, §1, §3

4. Bernstein, D.J., Lange, T., Niederhagen, R., Peters, C., Schwabe, P.: FSBday: implementing Wagner's generalized birthday attack against the SHA-3 round-1 candidate FSB. In: Indocrypt 2009 [23], pp. 18–38 (2009), http://eprint.iacr.org/2009/292, Citations in this document: §1
5. Bernstein, D.J., Lange, T., Peters, C.: Attacking and defending the McEliece cryptosystem. In: PQCrypto 2008 [9], pp. 31–46 (2008), http://eprint.iacr.org/2008/318, Citations in this document: §2, §4, §4
6. Bernstein, D.J., Lange, T., Peters, C.: Ball-collision decoding (2010), http://eprint.iacr.org/2010/585, Citations in this document: §2, §4
7. Bernstein, D.J., Lange, T., Peters, C., Schwabe, P.: Really fast syndrome-based hashing (2011), http://eprint.iacr.org/2011/074, Citations in this document: §1, §1, §5
8. Bernstein, D.J., Lange, T., Peters, C., van Tilborg, H.: Explicit bounds for generic decoding algorithms for code-based cryptography. In: WCC 2009, pp. 168–180 (2009), Citations in this document: §4
9. Buchmann, J., Ding, J. (eds.): PQCrypto 2008. LNCS, vol. 5299. Springer, Heidelberg (2008), See [5], [14]
10. Canteaut, A., Chabaud, F.: A new algorithm for finding minimum-weight words in a linear code: application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511. IEEE Transactions on Information Theory 44, 367–378 (1998), ftp://ftp.inria.fr/INRIA/tech-reports/RR/RR-2685.ps.gz, MR 98m:94043, Citations in this document: §4
11. Carleial, A.B., Hellman, M.E.: A note on Wyner's wiretap channel. IEEE Transactions on Information Theory 23, 387–390 (1977), ISSN 0018-9448, Citations in this document: §4
12. Wolfmann, J., Cohen, G. (eds.): Coding Theory 1988. LNCS, vol. 388. Springer, Heidelberg (1989), See [24]
13. Dawson, E., Vaudenay, S. (eds.): Mycrypt 2005. LNCS, vol. 3715. Springer, Heidelberg (2005), See [3]
14. Finiasz, M.: Syndrome based collision resistant hashing. In: PQCrypto 2008 [9], pp. 137–147 (2008), Citations in this document: §1
15. Finiasz, M., Gaborit, P., Sendrier, N.: Improved fast syndrome based cryptographic hash functions. In: Proceedings of ECRYPT Hash Workshop 2007 (2007), http://www-roc.inria.fr/secret/Matthieu.Finiasz/research/2007/finiasz-gaborit-sendrier-ecrypt-hash-workshop07.pdf, Citations in this document: §1, §1
16. Finiasz, M., Sendrier, N.: Security bounds for the design of code-based cryptosystems. In: Asiacrypt 2009 [20] (2009), http://eprint.iacr.org/2009/414, Citations in this document: §2, §4
17. Günther, C.G. (ed.): EUROCRYPT 1988. LNCS, vol. 330. Springer, Heidelberg (1988), MR 90a:94002, See [18]
18. Lee, P.J., Brickell, E.F.: An observation on the security of McEliece's public-key cryptosystem. In: Eurocrypt 1988 [17], pp. 275–280 (1988), http://dsns.csie.nctu.edu.tw/research/crypto/HTML/PDF/E88/275.PDF, MR 0994669, Citations in this document: §2, §4
19. Leon, J.S.: A probabilistic algorithm for computing minimum weights of large error-correcting codes. IEEE Transactions on Information Theory 34, 1354–1359 (1988), MR 89k:94072, Citations in this document: §2

20. Matsui, M. (ed.): ASIACRYPT 2009. LNCS, vol. 5912. Springer, Heidelberg (2009), See [16]
21. McEliece, R.J.: A public-key cryptosystem based on algebraic coding theory. JPL DSN Progress Report, 114–116 (1978), http://ipnpr.jpl.nasa.gov/progress_report2/42-44/44N.PDF, Citations in this document: §2
22. Prange, E.: The use of information sets in decoding cyclic codes. IRE Transactions on Information Theory IT-8, S5–S9 (1962), Citations in this document: §2
23. Roy, B., Sendrier, N. (eds.): INDOCRYPT 2009. LNCS, vol. 5922. Springer, Heidelberg (2009), See [4]
24. Stern, J.: A method for finding codewords of small weight. In: [12], pp. 106–113 (1989), Citations in this document: §2, §4

Ideal Secret Sharing Schemes for Useful Multipartite Access Structures

Oriol Farràs[1] and Carles Padró[2]

[1] Universitat Rovira i Virgili, Tarragona, Catalonia
[2] Nanyang Technological University, Singapore

Abstract. This paper is a survey of the main results and open problems in a line of work that was initiated shortly after secret sharing was introduced. Namely, the construction of ideal linear secret sharing schemes for access structures that are natural generalizations of the threshold ones and have interesting properties for the applications. Some of them have hierarchical properties, while other ones are suitable for situations requiring the agreement of several parties. These access structures are multipartite, that is, the participants are distributed into several parts and all participants in the same part play an equivalent role in the structure. This line of work has received an impulse from a recently discovered connection between ideal multipartite secret sharing schemes and integer polymatroids.

Keywords: Secret sharing, Ideal secret sharing schemes, Multipartite secret sharing, Hierarchical secret sharing, Integer polymatroids.

1 Introduction

Since its introduction in 1979 by Shamir [45] and Blakley [11], many different applications of secret sharing to several areas of Cryptology have appeared. In most applications, only threshold secret sharing schemes, as the ones in those seminal papers, are used. In addition, the homomorphic properties of Shamir's [45] threshold scheme make it suitable to be used in one of the main applications of secret sharing: secure multiparty computation [9,17,19]. Nevertheless, secret sharing for general (non-threshold) access structures has received a lot of attention. Two lines of research can be identified in the works on this topic.

The first one is the optimization of secret sharing schemes for general access structures. Most of the works on this line focus on two open problems. Namely, minimizing the length of the shares in relation to the length of the secret and the characterization of the access structures admitting an *ideal* secret sharing scheme, that is, a scheme in which all shares have the same length as the secret. These appeared to be extremely difficult open problems, with connections to several areas of Mathematics. Among the main results in this line of work we find the relation between ideal secret sharing and matroids discovered by Brickell and Davenport [15] and some subsequent findings about this connection [1,35,37,44,47], the proof that linear secret sharing schemes are not enough

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 99–108, 2011.
© Springer-Verlag Berlin Heidelberg 2011

to minimize the ratio between the length of the shares and the length of the secret [2,6,26], and the use of different combinatorial and information theoretical techniques [13,16,20,32,49] and, in particular, non-Shannon information inequalities [3,4] to find upper and lower bounds on the length of the shares.

The second line of research is more oriented towards the applications of secret sharing. It deals with constructions of ideal secret sharing schemes for *multipartite* access structures, in which the participants are distributed into several parts according to their role. These access structures are among the most natural generalizations of the threshold ones, and they are suitable for situations involving several parties as, for instance, hierarchical organizations. This was initiated by Kothari [34], Simmons [46], and Brickell [14], and it has been continued by several other authors [7,29,41,50,51].

This paper is a survey of the main results on that second line of research, with a special emphasis on the consequences of the recent introduction of integer polymatroids as a tool for the analysis and design of ideal multipartite secret sharing schemes [22].

2 Shamir's Threshold Secret Sharing Scheme

Since complete and detailed descriptions of Shamir's [45] threshold secret sharing scheme can be found in many texts (for instance in [48]), we only summarize here its main properties.

Shamir's scheme works for every (t, n)-*threshold access structure*, in which the qualified subsets are those having at least t out of n participants. The secret value is taken from a finite field with at least $n + 1$ elements. Since each share is taken from the same finite field as the secret, Shamir's scheme is ideal. In addition, it is *linear*, because both the generation of the shares and the secret reconstruction can be performed by computing values of some linear transformations. This implies homomorphic properties for Shamir's secret sharing scheme. Specifically, a linear combination of shares for different secrets result in shares for the corresponding linear combination of the secrets. Moreover, if the ratio between the number n of participants and the threshold t is large enough, Shamir's scheme has also homomorphic properties in relation to the multiplication in the finite field. Because of these multiplicative properties, it can be applied to the construction of secure multiparty computation protocols [9,17,19].

3 First Generalizations of Threshold Secret Sharing

The first secret sharing schemes for non-threshold access structures were introduced already in the seminal paper by Shamir [45], by modifying his threshold scheme to adapt it to situations in which some participants are more powerful than others. Specifically, every participant receives a certain number of shares from a threshold scheme. This scheme has a *weighted threshold access structure*, in which every participant has a weight and the qualified sets are those whose

weight sum attains the threshold. Since some shares are larger than the secret, this secret sharing scheme is not ideal.

Ito, Saito and Nishizeki [31] and Benaloh and Leichter [8] proved that there exists a secret sharing scheme for every access structure. Nevertheless, the size of the shares in those schemes grows exponentially with the number of participants. Actually, it is not possible to find an ideal scheme for every access structure [8], and in some cases the shares must be much larger than the secret [20]. Actually, the optimization of secret sharing schemes for general access structures has appeared to be an extremely difficult problem, and not much is known about it. Anyway, it seems clear that we cannot expect to find an efficient secret sharing scheme for every given access structure.

Nevertheless, this does not imply that efficient and useful secret sharing schemes only exist for threshold access structures. Actually, several constructions of ideal linear secret sharing schemes for access structures with interesting applications have been proposed.

Bloom [12] and Karnin, Greene and Hellman [33] presented alternative descriptions of Shamir's [45] and Blakley's [11] threshold schemes in terms of Linear Algebra. By generalizing the ideas in [12,33], Kothari [34] introduced the first ideal hierarchical secret sharing schemes.

Simmons [46] introduced two families of multipartite access structures, the so-called multilevel and compartmented access structures. The first ones are suitable for hierarchical organizations, while the second ones can be used in situations requiring the agreement of several parties. By generalizing the geometrical threshold scheme by Blakley [11], Simmons constructed ideal secret sharing schemes for some multilevel and compartmented access structures, and he conjectured that this was possible for all of them.

In a *multilevel access structure*, the participants are divided into m hierarchical levels and, for some given integers $0 < t_1 < \cdots < t_m$, a subset is qualified if and only if it has at least t_i participants in the first i levels for some $i = 1, \ldots, m$. A *compartmented access structure* is determined as well by some positive integers t and $t_1, \ldots, t_m$ with $t \geq \sum_{i=1}^{m} t_i$. The participants are divided into m compartments, and a subset is qualified if and only if it has at least t participants and, for every $i = 1, \ldots, m$, at least t_i participants in the i-th compartment.

4 Brickell's Ideal Secret Sharing Schemes

Simmons' [46] conjecture about the existence of ideal secret sharing schemes for the multilevel and the compartmented access structures was proved by Brickell [14]. This was done by introducing a new method, based on linear algebra, to construct ideal secret sharing schemes. This method has appeared to be very powerful and it has been used in most of the subsequent constructions of ideal secret sharing schemes. In addition, it provides a sufficient condition for an access structure to be *ideal*, that is, to admit an ideal scheme. This result was the first step in the discovery by Brickell and Davenport [15] of the connection between ideal secret sharing schemes and matroids.

We present the method by Brickell [14] to construct ideal secret sharing schemes as described by Massey [36] in terms of linear codes. Let C be an $[n+1, k]$-linear code over a finite field $\mathbb{K}$ and let M be a generator matrix of C, that is, a $k \times (n+1)$ matrix over $\mathbb{K}$ whose rows span C. Such a code defines an ideal secret sharing scheme on a set $P = \{p_1, \ldots, p_n\}$ of participants. Specifically, every random choice of a codeword $(s_0, s_1, \ldots, s_n) \in C$ corresponds to a distribution of shares for the secret value $s_0 \in \mathbb{K}$, in which $s_i \in \mathbb{K}$ is the share of the participant p_i. Such an ideal scheme is called a $\mathbb{K}$-*vector space secret sharing scheme* and its access structure is called a $\mathbb{K}$-*vector space access structure*.

It is easy to check that a set $A \subseteq P$ is in the access structure Γ of this scheme if and only if the column of M with index 0 is a linear combination of the columns whose indices correspond to the players in A. Therefore, if $Q = P \cup \{p_0\}$ and $\mathcal{M}$ is the representable matroid with ground set Q and rank function r that is defined by the columns of the matrix M, then

$$\Gamma = \Gamma_{p_0}(\mathcal{M}) = \{A \subseteq P \,:\, r(A \cup \{p_0\}) = r(A)\}.$$

That is, Γ is the *port of the matroid* $\mathcal{M}$ *at the point* p_0. Consequently, a sufficient condition for an access structure to be ideal is obtained. Namely, the ports of representable matroids are ideal access structures. Actually, they coincide with the vector space access structures. As a consequence the results by Brickell and Davenport [15], this sufficient condition is not very far from being necessary. Specifically, they proved that every ideal access structure is a matroid port. The reader is addressed to [35] for more information about matroid ports and their connection to ideal secret sharing.

By considering Reed-Solomon codes, one can check that Shamir's threshold scheme is a particular case of vector space secret sharing scheme. Like Shamir's threshold scheme, vector space secret sharing schemes are linear. Because of that, the algorithms to compute the shares and to recover the secret value are very efficient. In addition, linearity implies homomorphic properties of those schemes that are very useful for certain applications.

Brickell [14] proved that all multilevel and compartmented access structures are ideal. Specifically, he proved that, similarly to threshold access structures, every structure in one of those families admits a vector space secret sharing scheme over every large enough field. Even though the proof is constructive, it is still an open problem to determine how efficiently these schemes can be constructed.

One of the unsolved questions is to find out the minimum size of the fields over which there exist vector space secret sharing schemes for a given multilevel or compartmented access structure. This is an open problem as well for threshold access structures, equivalent to the main conjecture on maximum distance separable codes. Nevertheless, it is known that this minimum size of the field is linear on the number of participants for threshold structures, while the asymptotic behavior of this parameter is unknown for multilevel and compartmented access structures.

The computation time to construct a vector space secret sharing scheme for a multilevel or compartmented access structure is another open question. By

following the construction proposed by Brickell [14], a large number of determinants, which grows exponentially with the number of participants, have to be computed. An alternative method that avoids the necessity of computing this large number of determinants is proposed in the same work, but this construction is inefficient because it requires an extremely large field.

5 Constructing and Characterizing

The general method proposed by Brickell [14] is used in all subsequent constructions of ideal secret sharing schemes for multipartite access structures. Some of these works propose constructions for new families of multipartite access structures, while other papers deal with the aforementioned open problems about the efficiency of the constructions. In addition, the characterization of ideal multipartite access structures has attracted some attention as well. We describe some of these results in the following.

Tassa [50] considered a family of hierarchical access structures that are very similar to the multilevel ones. Specifically, given integers $0 < t_1 < \cdots < t_m$, a subset is qualified if and only if, for every $i = 1, \ldots, m$, it has at least t_i participants in the first i levels. Actually, these access structures are dual to the multilevel ones and, because of that, they admit as well vector space secret sharing schemes. The construction proposed by Tassa can be seen as a variant of Shamir's threshold scheme. As in Shamir's scheme, a random polynomial is used to determine the shares, but some of the shares are the values on some given points of the derivatives of certain orders instead of the values of the polynomial itself. The order of the derivative depends on the hierarchical level of the participant. Therefore, the secret value is reconstructed by using Birkhoff interpolation instead of Lagrange interpolation. Belenkiy [7] showed how to use Birkhoff interpolation to construct schemes for the multilevel access structures.

These constructions have the same efficiency problems as the ones by Brickell [14] for the multilevel and compartmented access structures. Nevertheless, Tassa proposes a probabilistic method that has a high practical interest. Specifically, one can estimate the probability of obtaining a matrix defining a secret sharing scheme with the required access structure when some of the parameters are chosen at random. This probability grows with the size of the field and it can be arbitrarily close to one.

Another construction of ideal multipartite secret sharing schemes is presented in [51]. In this case, polynomials on two variables are used. This construction is applied to the compartmented access structures and a variant of them and also to the family of hierarchical access structures considered in [50]. Constructions for other families of multipartite access structures are given in [29,41].

Other works deal with the characterization of the ideal access structures in some families of multipartite access structures. A complete characterization of the bipartite access structures that admit an ideal secret sharing scheme was presented by Padró and Sáez [42]. In particular, they characterized the ideal weighted threshold access structures with two weights. Other partial results

about the characterization of ideal weighted threshold access structures were given in [39], and this problem was completely solved by Beimel, Tassa, and Weinreb [5]. Partial results about the characterization of the ideal tripartite access structures were given in [18,29].

A common feature of all ideal access structures appearing in the works that have been surveyed in this section is that they admit vector space secret sharing schemes over every large enough field. In addition, the aforementioned open problems about the efficiency of the constructions of ideal schemes for multilevel and compartmented access structures appear as well for all those families.

In particular, determining the minimum size of the fields over which those structures admit a vector space secret sharing scheme has appeared to be extremely difficult. This problem is studied by Beutelspacher and Wettl [10] and by Giuletti and Vincenti [27] for particular cases of multilevel access structures. They present upper and lower bounds on the minimum size of the field for several multilevel access structures with two and three levels.

6 A New Tool: Integer Polymatroids

Integer polymatroids have been applied for the first time in [22] as a mathematical tool to study ideal multipartite secret sharing schemes. Specifically, this combinatorial object is used in that work to present a necessary condition and a sufficient condition for a multipartite access structure to be ideal. These results provide a general framework to analyze the previous constructions and characterizations, and also the existing open problems. We briefly describe them in the following, together with several important consequences that have been derived from them.

In the same way as matroids abstract some properties related to linear dependencies in collections of vectors in a vector space, integer polymatroids abstract similar properties in collections of subspaces of a vector space. Integer polymatroids have been thoroughly studied by researchers in combinatorial optimization, and the main results can be found in the books [25,40,43]. A concise presentation of the basic facts about integer polymatroids was given by Herzog and Hibi [30], who applied this combinatorial object to commutative algebra.

Brickell and Davenport [15] proved that every ideal secret sharing scheme with access structure Γ on a set P of participants defines a matroid $\mathcal{M}$ with ground set $Q = P \cup \{p_0\}$, such that Γ is the port of the matroid $\mathcal{M}$ at the point p_0. If the access structure Γ is m-partite, then the matroid $\mathcal{M}$ is $(m+1)$-partite. This is due to the fact that the symmetry properties of Γ are transported to the matroid $\mathcal{M}$, where one part consisting only of the point p_0 has to be added. Every $(m+1)$-partite matroid defines in a natural way an integer polymatroid on a ground set with $m+1$ elements. This implies the connection between ideal multipartite secret sharing schemes and integer polymatroids that is presented in [22]. In particular, a necessary condition for a multipartite access structure to be ideal is obtained.

In a similar way as some matroids can be represented by families of vectors, some integer polymatroids can be represented by families of vector subspaces.

One of the main results in [22] relates the representability of multipartite matroids and integer polymatroids. Specifically, a multipartite matroid is representable if and only if its associated integer polymatroid is representable. This provides a sufficient condition for a multipartite access structure to admit a vector space secret sharing scheme.

These general results about ideal multipartite access structures are applied in [22] to find a characterization of the ideal tripartite access structures. In addition, those results were used as well to find a characterization of the ideal hierarchical access structures [24]. As a consequence, a new proof for the characterization of the ideal weighted threshold access structures in [5] is obtained. It is proved in [24] that the ideal hierarchical access structures coincide with the hierarchical matroid ports and, moreover, that every hierarchical matroid port admits a vector space secret sharing scheme over every large enough field. The family of the tripartite access structures has the same properties.

As a consequence of the results in [22], if an integer polymatroid is representable over every large enough field, the same applies to the multipartite matroids that are associated to it. Therefore, every family of such integer polymatroids provides a family of multipartite access structures that admit vector space secret sharing schemes over every large enough field.

By analyzing the families of ideal multipartite access structures that have appeared in the literature under the light of the results in [22], we see that all of them are related to families of quite simple integer polymatroids. Of course, they are representable over every large enough field. For instance, the ideal bipartite and tripartite access structures, and also the compartmented ones, are obtained from integer polymatroids satisfying the strong exchange property [21]. On the other hand, all ideal hierarchical access structures are associated to Boolean polymatroids, which are very simple integer polymatroids that can be represented over every field (see [38] for more information). In particular, this applies to the multilevel access structures and to the ideal weighted threshold access structures.

7 Open Problems and Directions for Future Work

Unfortunately, the results in [22] do not solve the open problems related to the efficiency of the constructions of ideal secret sharing schemes for the multipartite access structures in those families. Nevertheless, those problems can be restated now in a clearer way. Specifically, as a consequence of [22, Theorem 6.1 (full version)], one should determine how efficiently a representation of a multipartite matroid can be obtained from a representation of its associated integer polymatroid. The proof of this theorem is constructive, but of course it does not provide the most efficient way to do that and, in addition, the given upper bound on the required size of the field is not tight.

Another direction for future work is to find new families of ideal multipartite access structures with similar properties as the ones analyzed in this paper. Namely, they should admit vector space secret sharing schemes over every large

enough finite field, and they should have additional properties that make them useful for the applications of secret sharing. By analogy to the previous families, it seems that one should analyze other families of simple integer polymatroids as, for instance, boolean polymatroids (only some of them define the ideal hierarchical access structures) or uniform integer polymatroids (see [23] for the definition).

Finally, characterizing the ideal access structures in other families of multipartite access structures is worth considering as well. For instance, one could try to characterize the ideal quadripartite access structures. Differently to the bipartite and tripartite cases, there exist quadripartite matroid ports that are not ideal. Namely, the access structures related to the Vamos Matroid. The results in [28] about the representability of integer polymatroids on four points can be very useful to obtain this characterization.

Acknowledgments and Disclaimer

The first author's work was partly funded by the Spanish Government through project CONSOLIDER INGENIO 2010 CSD2007-00004 "ARES", and by the Government of Catalonia through grant 2009 SGR 1135. The first author is with the UNESCO Chair in Data Privacy, but the views expressed in this paper are his own and do not commit UNESCO. The second author's work was supported by the Singapore National Research Foundation under Research Grant NRF-CRP2-2007-03.

References

1. Beimel, A., Chor, B.: Universally ideal secret-sharing schemes. IEEE Trans. Inform. Theory 40, 786–794 (1994)
2. Beimel, A., Ishai, Y.: On the power of nonlinear secret sharing schemes. SIAM J. Discrete Math. 19, 258–280 (2005)
3. Beimel, A., Livne, N., Padró, C.: Matroids Can Be Far from Ideal Secret Sharing. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 194–212. Springer, Heidelberg (2008)
4. Beimel, A., Orlov, I.: Secret Sharing and Non-Shannon Information Inequalities. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 539–557. Springer, Heidelberg (2009)
5. Beimel, A., Tassa, T., Weinreb, E.: Characterizing Ideal Weighted Threshold Secret Sharing. SIAM J. Discrete Math. 22, 360–397 (2008)
6. Beimel, A., Weinreb, E.: Separating the power of monotone span programs over different fields. SIAM J. Comput. 34, 1196–1215 (2005)
7. Belenkiy, M.: Disjunctive Multi-Level Secret Sharing. Cryptology ePrint Archive, Report 2008/018, `http://eprint.iacr.org/2008/018`
8. Benaloh, J.C., Leichter, J.: Generalized secret sharing and monotone functions. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 27–35. Springer, Heidelberg (1990)

9. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: Proc. ACM STOC 1988, pp. 1–10 (1988)
10. Beutelspacher, A., Wettl, F.: On 2-level secret sharing. Des. Codes Cryptogr. 3, 127–134 (1993)
11. Blakley, G.R.: Safeguarding cryptographic keys. In: AFIPS Conference Proceedings, vol. 48, pp. 313–317 (1979)
12. Bloom, J.R.: Threshold Schemes and Error Correcting Codes. Am. Math. Soc. 2, 230 (1981)
13. Blundo, C., De Santis, A., De Simone, R., Vaccaro, U.: Tight bounds on the information rate of secret sharing schemes. Des. Codes Cryptogr. 11, 107–122 (1997)
14. Brickell, E.F.: Some ideal secret sharing schemes. J. Combin. Math. and Combin. Comput. 9, 105–113 (1989)
15. Brickell, E.F., Davenport, D.M.: On the classification of ideal secret sharing schemes. J. Cryptology 4, 123–134 (1991)
16. Capocelli, R.M., De Santis, A., Gargano, L., Vaccaro, U.: On the size of shares of secret sharing schemes. J. Cryptology 6, 157–168 (1993)
17. Chaum, D., Crépeau, C., Damgård, I.: Multi-party unconditionally secure protocols. In: Proc. ACM STOC 1988, pp. 11–19 (1988)
18. Collins, M.J.: A Note on Ideal Tripartite Access Structures. Cryptology ePrint Archive, Report 2002/193, http://eprint.iacr.org/2002/193
19. Cramer, R., Damgård, I.B., Maurer, U.M.: General Secure Multi-party Computation from any Linear Secret-Sharing Scheme. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 316–334. Springer, Heidelberg (2000)
20. Csirmaz, L.: The size of a share must be large. J. Cryptology 10, 223–231 (1997)
21. Farràs, O.: Multipartite Secret Sharing Schemes. PhD Thesis, Universitat Politècnica de Catalunya (2010)
22. Farràs, O., Martí-Farré, J., Padró, C.: Ideal Multipartite Secret Sharing Schemes. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 448–465. Springer, Heidelberg (2007); The full version of this paper is available at the Cryptology ePrint Archive, Report 2006/292, http://eprint.iacr.org/2006/292
23. Farràs, O., Metcalf-Burton, J.R., Padró, C., Vázquez, L.: On the Optimization of Bipartite Secret Sharing Schemes. In: Kurosawa, K. (ed.) ICITS 2009. LNCS, vol. 5973, pp. 93–109. Springer, Heidelberg (2010)
24. Farràs, O., Padró, C.: Ideal hierarchical secret sharing schemes. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 219–236. Springer, Heidelberg (2010); The full version of this paper is available at the Cryptology ePrint Archive, Report 2009/141 (2010), http://eprint.iacr.org/2009/141
25. Fujishige, S.: Submodular Functions and Optimization. Annals of Discrete Mathematics, vol. 47. North-Holland Elsevier, Amsterdam (1991)
26. Gál, A.: A characterization of span program size and improved lower bounds for monotone span programs. In: Proceedings of 30th ACM Symposium on the Theory of Computing, STOC 1998, pp. 429–437 (1998)
27. Giuletti, M., Vincenti, R.: Three-level secret sharing schemes from the twisted cubic. Discrete Mathematics 310, 3236–3240 (2010)
28. Hammer, D., Romashchenko, A.E., Shen, A., Vereshchagin, N.K.: Inequalities for Shannon Entropy and Kolmogorov Complexity. J. Comput. Syst. Sci. 60, 442–464 (2000)
29. Herranz, J., Sáez, G.: New Results on Multipartite Access Structures. IEEE Proceedings on Information Security 153, 153–162 (2006)

30. Herzog, J., Hibi, T.: Discrete polymatroids. J. Algebraic Combin. 16, 239–268 (2002)
31. Ito, M., Saito, A., Nishizeki, T.: Secret sharing scheme realizing any access structure. In: Proc. IEEE Globecom 1987, pp. 99–102 (1987)
32. Jackson, W.-A., Martin, K.M.: Perfect secret sharing schemes on five participants. Des. Codes Cryptogr. 9, 267–286 (1996)
33. Karnin, E.D., Greene, J.W., Hellman, M.E.: On secret sharing systems. IEEE Trans. Inform. Theory 29, 35–41 (1983)
34. Kothari, S.C.: Generalized Linear Threshold Scheme. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 231–241. Springer, Heidelberg (1985)
35. Martí-Farré, J., Padró, C.: On Secret Sharing Schemes, Matroids and Polymatroids. J. Math. Cryptol. 4, 95–120 (2010)
36. Massey, J.L.: Minimal codewords and secret sharing. In: Proceedings of the 6-th Joint Swedish-Russian Workshop on Information Theory, Molle, Sweden, pp. 269–279 (August 1993)
37. Matúš, F.: Matroid representations by partitions. Discrete Math. 203, 169–194 (1999)
38. Matúš, F.: Excluded minors of Boolean polymatroids. Discrete Math. 253, 317–321 (2001)
39. Morillo, P., Padró, C., Sáez, G., Villar, J.L.: Weighted Threshold Secret Sharing Schemes. Inf. Process. Lett. 70, 211–216 (1999)
40. Murota, K.: Discrete convex analysis. SIAM Monographs on Discrete Mathematics and Applications. SIAM, Philadelphia (2003)
41. Ng, S.-L.: Ideal secret sharing schemes with multipartite access structures. IEEE Proc.-Commun. 153, 165–168 (2006)
42. Padró, C., Sáez, G.: Secret sharing schemes with bipartite access structure. IEEE Trans. Inform. Theory 46, 2596–2604 (2000)
43. Schrijver, A.: Combinatorial optimization. Polyhedra and efficiency. Springer, Berlin (2003)
44. Seymour, P.D.: On secret-sharing matroids. J. Combin. Theory Ser. B 56, 69–73 (1992)
45. Shamir, A.: How to share a secret. Commun. of the ACM 22, 612–613 (1979)
46. Simmons, G.J.: How to (Really) Share a Secret. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 390–448. Springer, Heidelberg (1990)
47. Simonis, J., Ashikhmin, A.: Almost affine codes. Des. Codes Cryptogr. 14, 179–197 (1998)
48. Stinson, D.R.: An explication of secret sharing schemes. Des. Codes Cryptogr. 2, 357–390 (1992)
49. Stinson, D.R.: Decomposition constructions for secret-sharing schemes. IEEE Transactions on Information Theory 40, 118–125 (1994)
50. Tassa, T.: Hierarchical Threshold Secret Sharing. J. Cryptology 20, 237–264 (2007)
51. Tassa, T., Dyn, N.: Multipartite Secret Sharing by Bivariate Interpolation. J. Cryptology 22, 227–258 (2009)

Loiss: A Byte-Oriented Stream Cipher*

Dengguo Feng, Xiutao Feng, Wentao Zhang, Xiubin Fan, and Chuankun Wu

State Key Laboratory of Information Security, Institute of Software,
Chinese Academy of Sciences, Beijing, 100190, China
{feng,fengxt,zhangwt,fxb,ckwu}@is.iscas.ac.cn

Abstract. This paper presents a byte-oriented stream cipher – Loiss, which takes a 128-bit initial key and a 128-bit initial vector as inputs, and outputs a keystream in bytes. The algorithm is based on a linear feedback shift register, and uses a structure called BOMM in the filter generator, which has good property on resisting algebraic attacks, linear distinguishing attacks and fast correlation attacks. In order for the BOMM to be balanced, the S-boxes in the BOMM must be orthomorphic permutations. To further improve the capability in resisting against those attacks, the S-boxes in the BOMM must also possess some good cryptographic properties, for example, high algebraic immunity, high nonlinearity, and so on. However current researches on orthomorphic permutations pay little attention on their cryptographic properties, and we believe that the proposal of Loiss will enrich the application of orthomorphic permutations in cryptography, and also motivate the research on a variety of cryptographic properties of orthomorphic permutations.

Keywords: stream cipher, Loiss, BOMM, orthomorphic permutation.

1 Introduction

Stream ciphers are widely used in secure network communications to protect communication data. A stream cipher algorithm is usually composed of a pseudorandom sequence generator and a plaintext mask function. The pseudorandom sequence generator first generates key streams under the control of an initial seed key, and then the plaintext mask function masks plaintexts with the above generated key streams and obtains the corresponding ciphertexts. Usually the mask function is the exclusive-OR operation.

Traditional stream cipher algorithms are mostly bit-oriented. With the rapid development of communication techniques, communication bandwidth becomes wider, and requirements on data throughput become higher. The traditional bit-oriented stream ciphers can hardly be designed to meet the requirements of communication applications in nowadays, specially in software implementations. For a more efficient use of modern computer hardware, some word-oriented (8/32-bit word) stream ciphers have been proposed, for example, SNOW 3G [1], and many algorithms submitted to the Europe eSTREAM project [2].

* This work was supported by the Natural Science Foundation of China (Grant No. 60833008 and 60902024) and the National 973 Program (Grant No. 2007CB807902).

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 109–125, 2011.
© Springer-Verlag Berlin Heidelberg 2011

A common design model of stream ciphers is to use a linear feedback shift register (LFSR) together with a nonlinear filter generator. The outputs of the LFSR go to the nonlinear filter generator for further process before the final keystream is produced. In this work we present a novel byte-oriented stream cipher – Loiss, which uses the above described model, and takes a 128-bit initial key and a 128-bit initial vector as inputs, and outputs a keystream in bytes. The Loiss algorithm has an LFSR, and uses a structure called byte-oriented mixer with memories (BOMM) as a part of the nonlinear filter generator. The BOMM itself contains some memory units and uses S-boxes as building blocks. The design of the BOMM component adopts the idea of the stream cipher RC4 [3], and adds the properties of confusion and accumulation. The BOMM has good capability in resisting against a number of common attacks on stream ciphers, including algebraic attack, linear distinguishing attack, and fast correlation attack. In order for the BOMM to be balanced in the statistical sense, the S-boxes in the BOMM must be orthomorphic permutations. What's more, to further improve the capability in resisting against those attacks, the S-boxes in the BOMM must also possess some good cryptographic properties, for example, high algebraic immunity, high nonlinearity, and so on. Unfortunately, little research results about these properties can be found from public literatures, and researches on orthomorphic permutations have been mainly about their construction and counting [5,6]. In the design of Loiss, extensive computing assistance together with some fundamental theoretical analysis are used. We believe that the proposal of Loiss will not only enrich the application of orthomorphic permutations in cryptography [7,8], and also motivate the research on cryptographic properties of orthomorphic permutations.

The rest of the paper is organized as follows: In section 2, we describe the Loiss algorithm in detail. And then we provide some basic properties of Loiss in section 3 and in-depth security analyses in section 4. In section 5 we give a simple analysis on software and hardware implementation cost of Loiss. Finally in section 6 we conclude the paper.

2 Description of Loiss

As stated above, the Loiss algorithm is a byte-oriented stream cipher, which generates a byte sequence (the keystream) under the control of a 128-bit initial key and a 128-bit initial vector. Loiss is logically composed of three parts: an LFSR, a nonlinear function F and a BOMM, see Figure 1.

2.1 The LFSR

Underlying finite field. Let F_2 be the binary field with elements 0 and 1 and $F_2[x]$ be the polynomial ring over F_2. The field F_{2^8} with 2^8 elements is defined

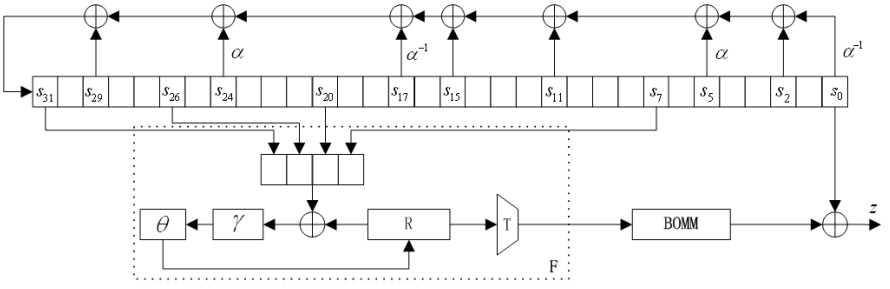


Fig. 1. The structure of Loiss

by a primitive polynomial $\pi(x) = x^8 + x^7 + x^5 + x^3 + 1$ over F_2 as the quotient $F_2[x]/(\pi(x))$. Let α be a root of the polynomial $\pi(x)$ in F_{2^8}, i.e., $\pi(\alpha) = 0$. Then $1, \alpha, \alpha^2, \cdots, \alpha^7$ form a basis of F_{2^8} and any element x in F_{2^8} can be written uniquely as

$$x = x_0 + x_1\alpha + \cdots + x_7\alpha^7,$$

where $x_i \in F_2$, $0 \le i \le 7$. Further the element x is represented by an 8-bit string or an 8-bit integer according to the following bijection mappings from F_{2^8} to the set $\{0,1\}^8$ or $\{0, 1, 2, \cdots, 2^8 - 1\}$:

$$x = \sum_{i=0}^{7} x_i\alpha^i \mapsto \quad x_7 \parallel x_6 \parallel \cdots \parallel x_0$$

or

$$x = \sum_{i=0}^{7} x_i\alpha^i \mapsto \quad \sum_{i=0}^{7} x_i 2^i,$$

where $\parallel$ denotes the concatenation of two bit strings. In this sense any element in F_{2^8} can be represented by an 8-bit string or an integer between 0 and 255.

Definition of the LFSR. The LFSR in the Loiss algorithm is defined over the field F_{2^8} and contains 32 byte registers, denote them as s_i, where $0 \le i \le 31$. The characteristic polynomial $f(x)$ of the LFSR is defined as below:

$$f(x) = x^{32} + x^{29} + \alpha x^{24} + \alpha^{-1}x^{17} + x^{15} + x^{11} + \alpha x^5 + x^2 + \alpha^{-1}. \qquad (1)$$

Let $(s_0^{(t)}, s_1^{(t)}, s_2^{(t)}, \cdots, s_{31}^{(t)})$ be the state of the LFSR at time t ($t \ge 0$). Then the state $(s_0^{(t+1)}, s_1^{(t+1)}, s_2^{(t+1)}, \cdots, s_{31}^{(t+1)})$ at time $t+1$ satisfies

$$s_{31}^{(t+1)} = s_{29}^{(t)} \oplus \alpha s_{24}^{(t)} \oplus \alpha^{-1}s_{17}^{(t)} \oplus s_{15}^{(t)} \oplus s_{11}^{(t)} \oplus \alpha s_5^{(t)} \oplus s_2^{(t)} \oplus \alpha^{-1}s_0^{(t)},$$
$$s_i^{(t+1)} = s_{i+1}^{(t)}, \quad i = 0, 1, 2, \cdots, 30.$$

2.2 The Nonlinear Function *F*

The nonlinear function F (the dotted rectangle in figure 1) is a compressing function from 32 bits to 8 bits, which contains a 32-bit memory unit R. The function F takes the values of the registers $s_{31}, s_{26}, s_{20}, s_7$ of the LFSR as inputs, and outputs a byte w, see Figure 2.

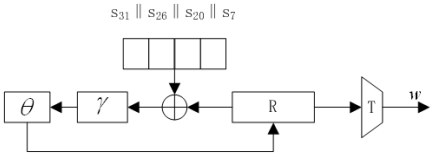


Fig. 2. The nonlinear function F

Let $s_{31}^{(t)}$, $s_{26}^{(t)}$, $s_{20}^{(t)}$ and $s_7^{(t)}$ be the values of the registers s_{31}, s_{26}, s_{20} and s_7 respectively at time t, and w be the output of F. Denote by $R^{(t)}$ and $R^{(t+1)}$ the values of the memory unit R at time t and $t+1$ respectively. Then we have

$$\begin{aligned} w &= T(R^{(t)}), \\ X &= s_{31}^{(t)} \parallel s_{26}^{(t)} \parallel s_{20}^{(t)} \parallel s_7^{(t)}, \\ R^{(t+1)} &= \theta(\gamma(X \oplus R^{(t)})), \end{aligned}$$

where $T(\cdot)$ is a truncation function which truncates the leftmost 8 bits from $R^{(t)}$ as output; γ is obtained by paralleling 4 S-boxes S_1 of size 8×8, that is,

$$\gamma(x_1 \parallel x_2 \parallel x_3 \parallel x_4) = S_1(x_1) \parallel S_1(x_2) \parallel S_1(x_3) \parallel S_1(x_4),$$

where x_i is a byte, $0 \le i \le 3$ and S_1 is defined in Table 4 (see appendix A); θ is a linear transformation on 32-bit strings, and is the same as the one used in the block cipher SMS4 [8], which is defined as

$$\theta(x) = x \oplus (x \lll 2) \oplus (x \lll 10) \oplus (x \lll 18) \oplus (x \lll 24), \tag{2}$$

where $\lll$ denotes the left cyclic shift on 32-bit strings.

2.3 The BOMM Structure

The BOMM is a transformation from 8 bits to 8 bits, and contains 16 byte memory units, denote them as y_i, $0 \le i \le 15$, see Figure 3.

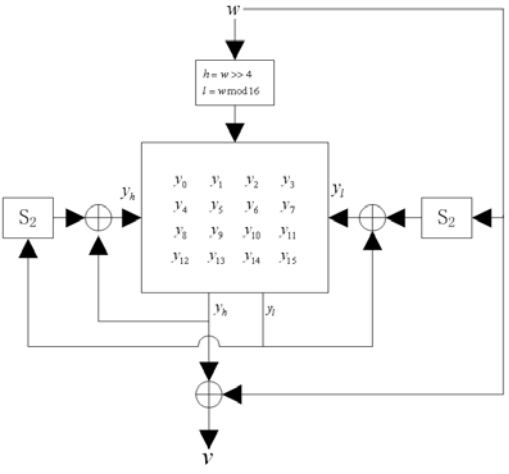


Fig. 3. The structure of BOMM

Let w and v be the input and the output of the BOMM respectively. Denote by $y_i^{(t)}$ and $y_i^{(t+1)}$ be the values of the memory units y_i at time t and $t+1$ respectively, where $i = 0, 1, \cdots, 15$. Then the update of memory cells in the BOMM is as follows:

$$
\begin{aligned}
&h = w \gg 4,\\
&l = w \bmod 16,\\
&\text{if } \ h \neq l \ \text{ then}\\
&\qquad y_l^{(t+1)} = y_l^{(t)} \oplus S_2(w),\\
&\qquad y_h^{(t+1)} = y_h^{(t)} \oplus S_2(y_l^{(t+1)}),\\
&\text{else}\\
&\qquad y_h^{(t+1)} = y_h^{(t)} \oplus S_2(w) \oplus S_2(y_h^{(t)} \oplus S_2(w)),\\
&\text{endif}\\
&y_i^{(t+1)} = y_i^{(t)}, \quad \text{for } i = 0, 1, \cdots, 15 \text{ and } i \neq h, l,
\end{aligned}
$$

where $\gg$ denotes the right shift operator, and S_2 is an S-box of size 8×8, see Table 5 in appendix A. The output of the BOMM is

$$v = y_h^{(t)} \oplus w. \tag{3}$$

2.4 Initialization of Loiss

The initialization process of Loiss can be divided into two stages:

In the first stage, it loads a 128-bit initial key and a 128-bit initial vector into the memory units of the LFSR and the BOMM as well, and then set the initial value of the 32-bit memory unit R of F to be zero, i.e., $R^{(0)} = 0$.

Set

$$\mathrm{IK} = \mathrm{IK}_0 \parallel \mathrm{IK}_1 \parallel \cdots \parallel \mathrm{IK}_{15},$$
$$\mathrm{IV} = \mathrm{IV}_0 \parallel \mathrm{IV}_1 \parallel \cdots \parallel \mathrm{IV}_{15},$$

where both IK_i and IV_i are bytes, $0 \le i \le 15$.

Let the initial states of the LFSR and the BOMM be $(s_0^{(0)}, s_1^{(0)}, \cdots, s_{31}^{(0)})$ and $(y_0^{(0)}, y_1^{(0)}, \cdots, y_{15}^{(0)})$ respectively. Then for $0 \le i \le 15$, we let

$$s_i^{(0)} = \mathrm{IK}_i,$$
$$s_{i+16}^{(0)} = \mathrm{IK}_i \oplus \mathrm{IV}_i,$$
$$y_i^{(0)} = \mathrm{IV}_i.$$

In the second stage, Loiss runs 64 times and the output of the BOMM takes part in the feedback calculation of the LFSR, see Figure 4.

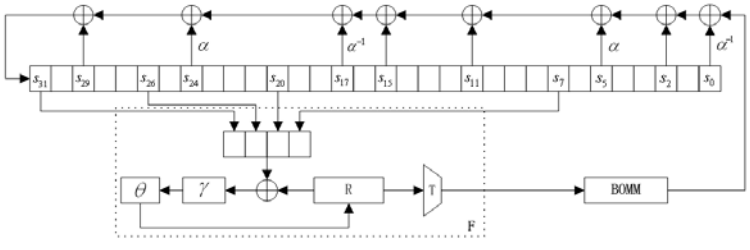


Fig. 4. The initialization of Loiss

2.5 Key Stream Generation

After the initialization, Loiss starts to produce key stream. Loiss produces one byte of key stream when it runs one time. Let z_t be the output of Loiss at time t. Then

$$z^{(t)} = s_0^{(t)} \oplus v^{(t)}, \tag{4}$$

where $s_0^{(t)}$ and $v^{(t)}$ are the value of the register s_0 of the LFSR and the output of the BOMM respectively at time t.

3 Some Basic Properties of the Components in Loiss

3.1 Properties of the LFSR

The LFSR of the Loiss algorithm is defined over the finite field F_{2^8}, and its characteristic polynomial $f(x)$ (see equation (1) for definition) is a primitive polynomial over F_{2^8} of degree 32. Thus non-zero sequences over F_{2^8} generated by $f(x)$ are m-sequences, and their periods are $2^{256} - 1$.

Let $\underline{a} = (a_0, a_1, \cdots, a_t, \cdots)$ be a non-zero sequence over F_{2^8} generated by $f(x)$. Note that $a_t \in F_{2^8}$ can be expressed as follows:

$$a_t = a_{t,7}\alpha^7 + a_{t,6}\alpha^6 + \cdots + s_{t,1}\alpha + a_{t,0}$$

for $t \geq 0$, where $1, \alpha, \cdots, \alpha^7$ is a basis of the finite field F_{2^8}. We call the sequence $\underline{a}_i = (a_{0,i}, a_{1,i}, \cdots, a_{t,i}, \cdots)$ $(0 \leq i \leq 7)$ derived from $\underline{a}$ as *the i-th coordinate sequence*. Then it is known [9] that each coordinate sequence of sequences generated by $f(x)$ is an m-sequence over the binary field F_2, and its characteristic polynomial is a primitive polynomial over F_2 of degree 256. And it is easy to verify that the characteristic polynomial of each coordinate sequence of non-zero sequences generated by $f(x)$ has a weight (the number of non-zero coefficients) of 131.

In addition, when choosing the polynomial $f(x)$, for a better resistance against linear distinguishing attack and fast correlation attack [10,11], we avoid as much as possible that $f(x)$ has some obvious multiples whose both degree and weight are low, and all coefficients are either 0 or 1.

3.2 Properties of *F*

The nonlinear function F contains a 32-bit memory unit R and uses S-boxes as building blocks. F itself has good resistance against linear distinguishing attack and fast correlation attack. Simple computation reveals that the following properties about F hold.

Property 1. *The algebraic degree, nonlinearity, differential uniformity and algebraic immunity of the S-box S_1 are 7, 112, 4 and 2 respectively.*

Property 2. *When we view θ as a transformation over the vector space $(F_{2^8})^4$, its differential and linear branch number are equal to 5.*

Let $\mathcal{L}$ be a linear approximation holding with probability p. Then the bias of $\mathcal{L}$ is defined as $\varepsilon = p - 1/2$. The following two properties are also easy to verify.

Property 3. *Supposed that the input sequence $\{X^{(t)}\}_{t\geq 0}$ of F and $R^{(0)}$ are independent and uniformly distributed. Then the bias of all linear approximations on inputs and outputs of 2-round F are zero.*

Property 4. *The number of active S-boxes of any linear approximations on inputs and outputs of 3-round F is at least 5.*

3.3 Properties of the BOMM

The S-box S_2

Property 5. *The algebraic degree, nonlinearity, differential uniformity and algebraic immunity of the S-box S_2 are 5, 112, 16 and 2 respectively.*

Definition 1. *Let $p(x)$ be a mapping over the finite field F_{2^n} with 2^n elements, where n is a positive integer. Then $p(x)$ is called an orthomorphic permutation if both $p(x)$ and $p(x) \oplus x$ are permutations over F_{2^n}.*

Property 6. *The S-box S_2 is an orthomorphic permutation over F_{2^8}.*

The balance of the BOMM. Let two random variables X and Y be independent and uniformly distributed, then for simplicity of description, X and Y are called IUD random variables.

Definition 2. *Suppose that the variables* $y_0^{(t)}$, $y_1^{(t)}$, $\cdots$, $y_{15}^{(t)}$ *and the input* w *of the BOMM are pairwise IUD random variables over* F_{2^8} *at time* t. *Denote by* v *the output of the BOMM at time* t. *Then the BOMM is called to be* balanced *if for arbitrary element* $a \in F_{2^8}$ *and* $0 \le i \le 15$, *we have*

$$\Pr(v = a) = \Pr\left(y_i^{(t+1)} = a\right) = \frac{1}{256}.$$

Property 7. *The BOMM is balanced if and only if the S-box* S_2 *is an orthomorphic permutation over* F_{2^8}.

Proof. We first prove the sufficiency. Let $h = w \gg 4$ and $l = w \bmod 16$. Then both h and l are IUD variables over the set $Z_{2^4} = \{0, 1, 2, \cdots, 15\}$. Since $v = y_h^{(t)} \oplus w$, and $y_h^{(t)}$ and w are IUD variables, it is easy to see that $\Pr(v = a) = \frac{1}{256}$ for arbitrary given $a \in F_{2^8}$. Now we consider the updates of the memory units y_k. Note that only two units y_l and y_h are updated at time t, thus we only need to prove that

$$\Pr\left(y_l^{(t+1)} = a\right) = \Pr\left(y_h^{(t+1)} = a\right) = \frac{1}{256}$$

for arbitrary given $a \in F_{2^8}$. Below we consider two cases.

1. $h \neq l$.
   Since $y_l^{(t+1)} = y_l^{(t)} \oplus S_2(w)$, note that $y_l^{(t)}$ and w are independent and S_2 is a permutation, thus we have

$$\begin{aligned}
&\Pr\left(y_l^{(t+1)} = a \mid h \neq l\right) = \Pr\left(y_l^{(t)} \oplus S_2(w) = a\right) \\
&= \sum_{b \in F_{2^8}} \Pr\left(y_l^{(t)} = b\right) \Pr\left(S_2(w) = a \oplus b\right) \\
&= 256 \cdot \frac{1}{256} \cdot \frac{1}{256} = \frac{1}{256}.
\end{aligned}$$

   Similarly, since $y_h^{(t+1)} = y_h^{(t)} \oplus S_2(y_l^{(t+1)})$, note that $y_h^{(t)}$ and $y_l^{(t+1)}$ are also independent, we can obtain $\Pr\left(y_h^{(t+1)} = a\right) = \frac{1}{256}$ in the same way.
2. $h = l$.
   Only one unit $y_h = y_l$ is updated in this case. Note that

$$y_h^{(t+1)} = y_h^{(t)} \oplus S_2(w) \oplus S_2(y_h^{(t)} \oplus S_2(w))$$

   and S_2 is an orthomorphic permutation, thus we have

$$\begin{aligned}
&\Pr\left(y_h^{(t+1)} = a \mid h = l\right) = \Pr\left(y_h^{(t)} \oplus S_2(w) \oplus S_2(y_h^{(t)} \oplus S_2(w)) = a\right) \\
&= \sum_{b \in F_{2^8}} \Pr\left(y_h^{(t)} \oplus S_2(w) = b\right) \Pr\left(S_2(b) \oplus b = a\right) \\
&= 256 \cdot \frac{1}{256} \cdot \frac{1}{256} = \frac{1}{256}.
\end{aligned}$$

Second we prove the necessity. At above we have proven that

$$\Pr\left(y_h^{(t+1)} = a \mid h \neq l\right) = \frac{1}{256}.$$

By the balance property of the BOMM, we have $\Pr\left(y_h^{(t+1)} = a \mid h = l\right) = \frac{1}{256}$. Since

$$\begin{aligned}
&\Pr\left(y_h^{(t+1)} = a \mid h = l\right) = \Pr\left(y_h^{(t)} \oplus S_2(w) \oplus S_2(y_h^{(t)} \oplus S_2(w)) = a\right) \\
&= \sum_{b \in F_{2^8}} \Pr\left(y_h^{(t)} \oplus S_2(w) = b\right) \Pr\left(S_2(b) \oplus b = a\right) \\
&= \frac{1}{256} \sum_{b \in F_{2^8}} \Pr\left(S_2(b) \oplus b = a\right),
\end{aligned}$$

thus we have $\sum_{b \in F_{2^8}} \Pr\left(S_2(b) \oplus b = a\right) = 1$ for any $a \in F_{2^8}$. It follows that the equation $S_2(x) \oplus x = a$ has exactly one solution for any $a \in F_{2^8}$, that is, $S_2(x) \oplus x$ is a permutation. So S_2 is an orthomorphic permutation. ■

3.4 Key Entropy Preservation in the Initialization

In Loiss the process of the initialization preserves the entropy of keys, namely, there is a one-to-one mapping from the initial state of Loiss to the state after its initialization.

Property 8. *Denote $IS(t)$ by the internal state of Loiss at time t, that is, $IS(t) = (s_0^{(t)}, \cdots, s_{31}^{(t)}, R^{(t)}, y_0^{(t)}, \cdots, y_{15}^{(t)})$. Then $IS(t)$ is determined by $IS(t+1)$.*

Proof. Suppose that $IS(t+1) = (s_0^{(t+1)}, \cdots, s_{31}^{(t+1)}, R^{(t+1)}, y_0^{(t+1)}, \cdots, y_{15}^{(t+1)})$ is known. By $IS(t+1)$, we have

- By the shift of the LFSR, namely $s_{i-1}^{(t+1)} = s_i^{(t)}$, we recover $s_i^{(t)}$ for $i = 1, 2, \cdots, 31$;
- Let $X = s_{31}^{(t)} \parallel s_{26}^{(t)} \parallel s_{20}^{(t)} \parallel s_7^{(t)}$. Note that both θ and γ are bijective, by the update of R, when X and $R^{(t+1)}$ are known, we recover $R^{(t)}$;
- Let $w = T(R^{(t)})$, $h = w >> 4$ and $l = w \bmod 16$. By the update of the BOMM, we recover all $y_i^{(t)}$ by $y_i^{(t+1)}$, where $0 \leq i \leq 15$ and $i \neq h, l$. When $h \neq l$, we have $y_l^{(t)} = y_l^{(t+1)} \oplus S_2(w)$ and $y_h^{(t)} = y_h^{(t+1)} \oplus S_2(y_l^{(t+1)})$. When $h = l$, let $u = y_h^{(t)} \oplus S_2(w)$. Note that the S-box S_2 is an orthomorphic permutation, there is exactly one solution u in equality $y_h^{(t+1)} = u \oplus S_2(u)$. We denote by u such a solution. So $y_h^{(t)} = u \oplus S_2(w)$. Up to now we have recovered all values of memory cells of the BOMM.
- Let $v = y_h^{(t)} \oplus w$. By the feedback of the LFSR, we have

$$s_0^{(t)} = \alpha(s_{31}^{(t+1)} \oplus s_{29}^{(t)} \oplus \alpha s_{24}^{(t)} \oplus \alpha^{-1} s_{17}^{(t)} \oplus s_{15}^{(t)} \oplus s_{11}^{(t)} \oplus \alpha s_5^{(t)} \oplus s_2^{(t)} \oplus v).$$

So $IS(t)$ is determined by $IS(t+1)$. ■

4 Security Analysis

4.1 Guess and Determine Attack

The guess and determine attack is a known plaintext attack for state recovery [12,13]. Its main idea is that: an attacker first guesses the value of a portion of inner states of the target algorithm, then it takes a little cost to deduce all the rest of the inner states making use of the guessed portion of the inner states and a few known key stream bits.

As for Loiss, below we construct a guess and determine attack, which needs the outputs of F at the 7 successive times.

Definition 3. *Let $w^{(t+i)}$ be the outputs of the nonlinear function F at the seven successive times starting from time t, $0 \le i \le 6$. Then when the following conditions are met, we call it an event* **A***:*

1. $h^{(t)} = l^{(t)}$, *where* $h^{(t)} = w^{(t)} \gg 4$ *and* $l^{(t)} = w^{(t)} \bmod 16$;
2. $w^{(t)} = w^{(t+1)} = \cdots = w^{(t+6)}$.

When the event **A** occurs at time t, the attacker guesses the values of $h^{(t)}$, $s_0^{(t)}$, $s_{12}^{(t)}$, $s_{14}^{(t)}$, $s_{15}^{(t)}$, $s_{25}^{(t)}$ and the values of the rightmost three bytes of each $R^{(t+i)}$, $0 \le i \le 5$, then recovers the values of all the rest inner states of the LFSR and F. After all inner states of the LFSR and F are recovered, the attacker runs Loiss for about another 128 ($= 2^7$) times and then can recover the values of all memory units of the BOMM. Since the probability that the event **A** occurs is 2^{-52} and the attacker has to guess 188-bit inner states in the guessing stage, so the time complexity of the above attack method is $O(2^{247})$ and its data complexity is $O(2^{52})$.

4.2 Linear Distinguishing Attack

The linear distinguishing attack is a common attack on stream ciphers [14,15]. Its basic idea is that: an attacker first constructs a linear distinguisher by means of linear approximations of the nonlinear part of an algorithm, and the linear distinguisher only depends on the key stream. When the bias of constructed linear distinguisher is significant, the attacker can distinguish key streams generated by the algorithm from a true random bit stream by means of the above distinguisher.

Now we consider the linear distinguishing attack on Loiss. The nonlinear part of Loiss includes F and the BOMM. First we consider linear approximations of F. Assume that the inputs of F and the value of the memory unit R are IDU random variables within a short successive time interval. Since the linear branch number of the transformation θ is 5, thus the bias of arbitrary linear approximations of 2-round F is zero. Below we construct a linear approximation of 3-round F:

$$a \cdot (w^{(t)} \oplus s_{31}^{(t)}) \oplus c \cdot X^{(t+1)} \oplus b \cdot w^{(t+2)} = 0, \tag{5}$$

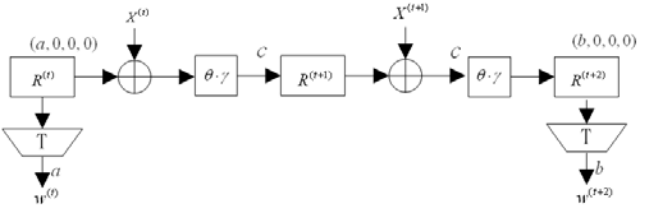


Fig. 5. Linear approximations of 3-round F

where a and b are 8-bit strings, c is a 32-bit string, and "$\cdot$" denotes the inner product of bit strings. The rationale of the linear approximation (5) is demonstrated in Figure 5.

Going through all possible values of a, b and c, we can obtain that the maximum bias of the linear approximation (5) is $2^{-16.4}$.

Second we consider linear approximations of the BOMM. Below we only consider the inputs $w^{(t)}, w^{(t+2)}$ and the outputs $v^{(t)}, v^{(t+2)}$ of the BOMM at times t and $t+2$.

Definition 4. *For the inputs $w^{(t)}, w^{(t+2)}$ of the BOMM at times t and $t+2$, if the equalities $h^{(t)} = l^{(t)} = h^{(t+2)} = h$ hold, where $h^{(t)} = w^{(t)} \gg 4$, $l^{(t)} = w^{(t)} \bmod 16$, $h^{(t+2)} = w^{(t+2)} \gg 4$, and the memory unit y_h is not updated at time $t+1$, then we call it an event* **B**.

By definition, the probability that the event **B** occurs is $\Pr(B) = 2^{-8}\left(\frac{15}{16}\right)^2$. When the event **B** occurs, we have

$$v^{(t)} \oplus v^{(t+2)} \oplus w^{(t)} \oplus w^{(t+2)} = S_2(w^{(t)}) \oplus S_2(v^{(t)} \oplus w^{(t)} \oplus S_2(w^{(t)})). \quad (6)$$

And we can construct the following linear approximation of equation (6):

$$a \cdot w^{(t)} \oplus b \cdot w^{(t+2)} = d \cdot v^{(t)} \oplus e \cdot v^{(t+2)}, \quad (7)$$

where a, b, d and e are 8-bit strings.

Going through all possible values of a, b, d and e, we have that the maximum bias of the linear approximation (7) is $2^{-15.2}$.

Combining the linear approximations (5) and (7), and going through all possible values of a, b, c, d and e, by the Piling-up Lemma [16] we obtain that the maximum bias of linear approximations of the nonlinear part of Loiss is $2^{-30.6}$, at the same time the bias of linear approximations of both F and the BOMM reach the maximum possible value, that is, $2^{-16.4}$ and $2^{-15.2}$ respectively.

Suppose that an attacker has got a trinomial multiple of the characteristic polynomial $f(x)$ of the LFSR with low degree whose all non-zero coefficients are one. Then by means of the above linear approximations, the attacker can construct a only key stream linear distinguisher with the maximum bias, whose bias is about $2^{3-1}(2^{-30.6})^3 = 2^{-89.8}$. By means of this linear distinguisher the attacker needs about 2^{180}-bit key stream to distinguish key streams generated

by Loiss from a true random bit stream [16]. In fact, it is yet an open problem whether the attacker can obtain such a trinomial multiple with low degree whose all non-zero coefficients are one, which might not exist. Thus the above data complexity is in an optimistic case. This shows that Loiss has a good resistance against linear distinguishing attacks.

4.3 Algebraic Attacks

The algebraic attack is a powerful attack to cryptosystems [17,18,19]. Its main idea is that: a cryptographic system can be viewed as an algebraic equation system, and then the problem about breaking the cryptographic system can be converted into the problem of solving the corresponding algebraic equation system.

As for Loiss, we will consider how to establish such an algebraic equation system and give a simple estimation on the time complexity of solving it.

First we consider F whose nonlinear part is only the S-box S_1. When F runs one time, the S-box S_1 will be called for four times. Since the algebraic immunity of S_1 is 2, and at most 39 linearly independent quadratic equations on the inputs and outputs of S_1 can be established, thus at most $39 \times 4 = 156$ linearly independent quadratic equations can be established when F runs for one time.

Second we consider the BOMM. Here we introduce choosing functions $h_i(w)$ and $l_i(w)$, where $h_i(w)$ and $l_i(w)$ are boolean functions from F_{2^8} to F_2 and defined as follows:

$$h_i(w) = \begin{cases} 1, \text{ if } w \gg 4 = i, \\ 0, \text{ otherwise,} \end{cases}$$

$$l_i(w) = \begin{cases} 1, \text{ if } w \bmod 16 = i, \\ 0, \text{ otherwise,} \end{cases}$$

where $0 \le i \le 15$, and the intermediate variables $yl^{(t)}$, $yh^{(t)}$, $Sl^{(t)}$ and $Sh^{(t)}$ satisfying

$$yl^{(t)} = y_0^{(t)} \cdot l_0(w^{(t)}) \oplus y_1^{(t)} \cdot l_1(w^{(t)}) \oplus \cdots \oplus y_{15}^{(t)} \cdot l_{15}(w^{(t)}), \tag{8}$$

$$yh^{(t)} = y_0^{(t)} \cdot h_0(w^{(t)}) \oplus y_1^{(t)} \cdot h_1(w^{(t)}) \oplus \cdots \oplus y_{15}^{(t)} \cdot h_{15}(w^{(t)}), \tag{9}$$

$$Sl^{(t)} = S_2(w^{(t)}), \tag{10}$$

$$Sh^{(t)} = S_2(yl^{(t)} \oplus Sl^{(t)}). \tag{11}$$

Then we can establish the update function of the BOMM as follows:

$$y_i^{(t+1)} = y_i^{(t)} \oplus Sl^{(t)} l_i(w^{(t)}) \oplus Sh^{(t)} h_i(w^{(t)}), 0 \le i \le 15. \tag{12}$$

Note that the boolean functions $h_i(w)$ and $l_i(w)$ have an algebraic degree of 4, thus the algebraic degree of equations (8), (9) and (12) are 5. Therefore when the

BOMM updates for one time, we can obtain 144 equations of degree 5. Note that each time when the BOMM is updated, the S-box S_2 is invoked for two times, see equations (10) and (11), since the algebraic immunity of S_2 is 2, and at most 36 linearly independent quadratic equations can be established, thus we can obtain 72 linearly independent quadratic equations. Finally by the output equation $v^{(t)} = yh^{(t)} \oplus w^{(t)}$ of the BOMM, we can obtain 8 linear equations. Totally during one time update of the BOMM we obtain 72 linearly independent quadratic equations and 144 equations of degree 5, and introduce 152 ($= 3 \times 8 + 16 \times 8$) independent intermediate bit variables.

Assume that the inner states of Loiss at time t are IDU random variables. Since Loiss has 416 bits of inner states (LFSR:256+F:32+BOMM:128), an attacker needs at least 52 (=416/8) byte of key stream to possibly establish an over-defined algebraic equation system. Similarly to the above process, we can establish an entire algebraic equation system for Loiss with 52 key stream bytes, which contains 9800 variables (including introduced intermediate variables) and 18980 equations. The highest degree of the entire algebraic equation system is 5.

When applying the normal linearization to solve the above algebraic equation system, the number of linear equations is much less than that of variables because the linearization process introduces many more intermediate variables. Therefore the normal linearization method cannot be directly used to solve it. Here we use the XL method [20] to estimate the time complexity of solving the above algebraic equation system, and obtain that its time complexity is about $2^{2420.88}$, which is much higher than the time complexity of a brute force attack. Therefore Loiss has good resistance against algebraic attacks.

4.4 Time-Memory-and-Data Attack

The time-memory-and-data attack is a basic method in computer science [21,22,23,24]. Its main idea is that: reduce the cost of space by sacrifice the cost in time, or vice versa. In analyzing stream ciphers, the data is also taken into consideration.

More precisely, denote by D, T and M respectively the number of data (plaintext-ciphertext pairs) an attacker can get, the time complexity of the attack, and the size of memory required to perform an attack. Then D, T and M satisfy $TM^2D^2 = N^2$ and $N > T \geq D^2$, where N denote the size of the space of unknowns that the attack targets to recover. Normally we assume the number of data got by an attacker reaches the upper bound, that is, $T = D^2$, then we have $TM = N$.

With respect to the time-memory-and-data attack, since Loiss contains a total of 416 bits of unknowns, thus $N = 2^{416}$. It follows that $TM = 2^{416}$. This shows that at least one of T and M is no less than 2^{208}. So Loiss have good resistance against the time-memory-and-data attack.

5 Evaluations on Software and Hardware Implementations

5.1 On Software Implementation

Since Loiss's basic operators are byte-oriented, thus a software implementation can make it very fast, and the software implementation only needs small memory and small size of code, and hence the algorithm can be used in resource limited environments, e.g., in smart cards. Compared to the well-known block cipher AES [4], the encryption speed of Loiss is almost the same as that of 128-AES (whose keys has a length of 128) in the counter mode. Below we give a simple performance evaluation on software implementations on a 32-bit common PC according to the Intel 486 32-bit instruction set, and compare it with that of SNOW 3G, see Table 1.

Table 1. Performance of software implementation of parts of Loiss (Unit: Cycles)

Algorithm Name	LFSR	F	BOMM	Initialization	Generate single key
Loiss	13	21	14	3223	48
SNOW 3G	14	31		1508	46

Note: From the above table it can be seen that the speed of generating single key by Loiss is almost the same as that of SNOW 3G. But since SNOW 3G generates a 32-bit word one time, thus totally the speed of generating keystream by SNOW 3G is three times faster than the one by Loiss in common length.

Table 2. Size of electric circuits in hardware implementation of Loiss

Units	Num. of units	Num. of Gates
8-bit register	48	$8 \times 48 \times 10 = 3840$
32-bit register	1	$32 \times 1 \times 10 = 320$
multiplication by α and α^{-1}	4	$10 \times 4 = 40$
S-boxes	6	$500 \times 6 = 3000$
8-bit XOR	12	$22 \times 12 = 264$
32-bit XOR	5	$86 \times 5 = 430$
two-choose-one logic	5	$8 \times 5 = 40$
total	7934	

5.2 On Hardware Implementation

There are different approaches in hardware implementation. As an approach of hardware implementation of Loiss, we give a rough estimation on the size of electric circuits needed in implementing each part of Loiss, where the number of gates can further be optimized, see Table 2. In addition, we also give a simple comparison of the sizes of electric circuits in hardware implementations of both Loiss and SNOW 3G, see Table 3.

Table 3. Comparison of sizes of electric circuits in hardware implementation of Loiss and SNOW 3G

	Loiss	SNOW 3G
Size	7934	10908

6 Conclusions

In this paper we present a byte-oriented stream cipher Loiss. The Loiss algorithm has good performance of software and hardware implementations, and is suitable for a variety of software and hardware implementation requirements. Loiss has good properties in resisting against many known attacks, including guess and determine attack, linear distinguishing attack, algebraic attack, time-memory-and-data attack, and fast correlation attack, and can offer the 128-bit-level security. In the design of Loiss, an orthomorphic permutation is used, which is necessary to ensure the balance of the BOMM component, and it is expected to motivate the research on the cryptographic properties of orthomorphic permutations.

Acknowledgement

During the design of Loiss, a large number of graduate students from the State Key Laboratory of Information Security, Chinese Academy of Sciences, have made significant contributions. Those students are highly appreciated.

References

1. ETSI/SAGE, SNOW 3G Specification, Specification of the 3GPP Confidentiality and Integrity Algorithms UEA2 & UIA2, Document 2 (September 2006)
2. eSTREAM, ECRYPT Stream Cipher Project, `http://www.ecrypt.eu.org/stream`
3. Rivest, R.L.: The RC4 encryption algorithm, RSA Data Security, Inc. (March 1992)
4. FIPS PUB 197, The official AES standard, `http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf`
5. Mittenthal, L.: Block substitutions using orthomorphic mappings. Advances in Applied Mathematics 16(1), 59–71 (1995)

6. Lv, S.W., Fan, X.B., Wang, Z.S., Xu, J.L., Zhang, J.: Completing mappings and their appliactions. University of Sciences and Technology of China Press (2008)
7. Vaudenay, S.: On the Lai-Massey Scheme. In: Lam, K.-Y., Okamoto, E., Xing, C. (eds.) ASIACRYPT 1999. LNCS, vol. 1716, pp. 8–19. Springer, Heidelberg (1999)
8. Chinese State Bureau of Cryptography Administration, Cryptographic algorithms SMS4 used in wireless LAN products, `http://www.oscca.gov.cn/Doc/6/News_1106.htm`
9. Golomb, S.W., Gong, G.: Signal design for good correlation for wireless communication, cryptography and radar. Cambridge University Press, Cambridge (2004)
10. Zeng, K., Huang, H.: On the linear syndrome method in cryptanalysis. In: EUROCRYPT 1988, pp. 469–478 (1990)
11. Siegenthaler, T.: Correlation-immunity of nonlinear combining functions for cryptographic applications. IEEE Transaction on Information Theory, IT-30, 776–780 (1984)
12. Canniere, C.: Guess and determine attack on SNOW, NESSIE Public Document, NES/DOC/KUL/WP5/011/a (2001)
13. Hawkes, P., Rose, G.G.: Guess-and-Determine Attacks on SNOW. In: Nyberg, K., Heys, H.M. (eds.) SAC 2002. LNCS, vol. 2595, pp. 37–46. Springer, Heidelberg (2003)
14. Watanabe, D., Biryukov, A., Canniere, C.: A distinguishing attack of SNOW 2. In: Matsui, M., Zuccherato, R.J. (eds.) SAC 2003. LNCS, vol. 3006, pp. 222–233. Springer, Heidelberg (2004)
15. Coppersmith, D., Halevi, S., Jutla, C.S.: Cryptanalysis of Stream Ciphers with Linear Masking. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 515–532. Springer, Heidelberg (2002)
16. Matsui, M.: Linear cryptanalysis method for DES cipher. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 386–397. Springer, Heidelberg (1994)
17. Courtois, N.T., Meier, W.: Algebraic attacks on stream ciphers with Linear Feedback. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 346–359. Springer, Heidelberg (2003)
18. Ronjom, S., Helleseth, T.: Attacking the filter generator over GF(2^m). In: Workshop Record of SASC 2007: The State of the Art of Stream Ciphers, eSTREAM report 2007/011 (2007)
19. Meier, W., Pasalic, E., Carlet, C.: Algebraic Attacks and Decomposition of Boolean Functions. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 474–491. Springer, Heidelberg (2004)
20. Diem, C.: The XL-Algorithm and a Conjecture from Commutative Algebra. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 323–337. Springer, Heidelberg (2004)
21. Hellman, M.E.: A cryptanalytic time-memory tradeoff. IEEE Transactions on Information Theory 26, 401–406 (1980)
22. Biryukov, A., Shamir, A.: Cryptanalytic time/Memory/Data tradeoffs for stream ciphers. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 1–13. Springer, Heidelberg (2000)
23. Biryukov, A., Mukhopadhyay, S., Sarkar, P.: Improved time-memory trade-offs with multiple data. In: Preneel, B., Tavares, S. (eds.) SAC 2005. LNCS, vol. 3897, pp. 110–127. Springer, Heidelberg (2006)
24. Hong, J., Sarkar, P.: New Applications of Time Memory Data Tradeoffs. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 353–372. Springer, Heidelberg (2005)

Appendix A: The S-Boxes S_1 and S_2

For an S-box S of size 8×8 which can be S_1 or S_2, let $x \in F_{2^8}$ and $h = x \gg 4$ and $l = x \bmod 16$. Then $S(x)$ is the element at the intersection of the h-th row and the l-th column in Tables 4 or 5. For example, $S_1(0x3A) = 0xBF$.

Note: Data in Table 4 and 5 are expressed in the hexadecimal format.

Table 4. The S-box S_1

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	55	C2	63	71	3B	C8	47	86	9F	3C	DA	5B	29	AA	FD	77
1	8C	C5	94	0C	A6	1A	13	00	E3	A8	16	72	40	F9	F8	42
2	44	26	68	96	81	D9	45	3E	10	76	C6	A7	8B	39	43	E1
3	3A	B5	56	2A	C0	6D	B3	05	22	66	BF	DC	0B	FA	62	48
4	DD	20	11	06	36	C9	C1	CF	F6	27	52	BB	69	F5	D4	87
5	7F	84	4C	D2	9C	57	A4	BC	4F	9A	DF	FE	D6	8D	7A	EB
6	2B	53	D8	5C	A1	14	17	FB	23	D5	7D	30	67	73	08	09
7	EE	B7	70	3F	61	B2	19	8E	4E	E5	4B	93	8F	5D	DB	A9
8	AD	F1	AE	2E	CB	0D	FC	F4	2D	46	6E	1D	97	E8	D1	E9
9	4D	37	A5	75	5E	83	9E	AB	82	9D	B9	1C	E0	CD	49	89
A	01	B6	BD	58	24	A2	5F	38	78	99	15	90	50	B8	95	E4
B	D0	91	C7	CE	ED	0F	B4	6F	A0	CC	F0	02	4A	79	C3	DE
C	A3	EF	EA	51	E6	6B	18	EC	1B	2C	80	F7	74	E7	FF	21
D	5A	6A	54	1E	41	31	92	35	C4	33	07	0A	BA	7E	0E	34
E	88	B1	98	7C	F3	3D	60	6C	7B	CA	D3	1F	32	65	04	28
F	64	BE	85	9B	2F	59	8A	D7	B0	25	AC	AF	12	03	E2	F2

Table 5. The S-box S_2

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	61	97	FF	E9	66	56	F1	F3	54	72	CC	4D	85	52	7A	70
1	D0	2E	4C	58	BE	88	7F	5A	2F	1B	47	AF	9B	D5	BF	81
2	C3	4E	86	2D	6A	9C	CE	20	2B	53	6D	FD	3C	BC	33	22
3	F7	59	C9	63	6E	8D	DD	F2	E3	1A	75	DA	13	1D	68	42
4	A4	3F	B7	46	90	12	73	EB	FA	F6	09	40	A5	E0	B4	B1
5	51	8E	06	34	7D	DF	99	6F	AA	0B	80	95	25	EA	87	CD
6	DC	0C	43	FB	A7	BD	9E	FC	EE	9F	74	B6	CF	EF	16	0F
7	78	D1	92	64	D6	84	48	41	08	60	5D	2A	B8	4F	E2	69
8	01	C1	31	5F	62	49	B2	93	00	CB	04	18	07	71	17	E4
9	AC	8B	B0	7E	F8	44	5B	AD	98	A0	27	4B	3A	B5	F0	83
A	F9	14	E7	23	77	D2	10	AE	B3	36	30	3B	1C	03	82	38
B	0E	7B	50	A6	1F	7C	CA	C2	02	2C	A9	8A	39	15	F4	D9
C	A3	55	32	96	C8	8C	C0	05	67	1E	EC	19	29	89	F5	21
D	37	BB	E1	57	A2	C7	E6	8F	AB	91	35	28	D3	D7	79	BA
E	A1	6C	B9	DE	A8	5E	FE	6B	C5	ED	65	9A	45	C6	C4	9D
F	94	24	0D	0A	E5	76	3D	E8	26	5C	D4	4A	D8	11	DB	3E

Secure Message Transmission by Public Discussion: A Brief Survey

Juan Garay[1], Clint Givens[2], and Rafail Ostrovsky[3,⋆]

[1] AT&T Labs – Research
garay@research.bell-labs.com
[2] Department of Mathematics, UCLA
cgivens@math.ucla.edu
[3] Departments of Computer Science and Mathematics, UCLA
rafail@cs.ucla.edu

Abstract. In the problem of Secure Message Transmission in the public discussion model (SMT-PD), a Sender wants to send a message to a Receiver privately and reliably. Sender and Receiver are connected by n channels, up to $t < n$ of which may be maliciously controlled by a computationally unbounded adversary, as well as one public channel, which is reliable but not private. The SMT-PD abstraction has been shown instrumental in achieving secure multi-party computation on sparse networks, where a subset of the nodes are able to realize a broadcast functionality, which plays the role of the public channel.

In this short survey paper, after formally defining the SMT-PD problem, we overview the basic constructions starting with the first, rather communication-inefficient solutions to the problem, and ending with the most efficient solutions known to-date—optimal private communication and sublinear public communication.

These complexities refer to resource use for a single execution of an SMT-PD protocol. We also review the *amortized* complexity of the problem, which would arise in natural use-case scenarios where $\mathcal{S}$ and $\mathcal{R}$ must send several messages back and forth, where later messages depend on earlier ones.

1 Introduction and Motivation

The model of *Secure Message Transmission* (SMT) was introduced by Dolev, Dwork, Waarts and Yung [DDWY93] in an effort to understand the connectivity requirements for secure communication in the information-theoretic setting. Generally speaking, an SMT protocol involves a sender, $\mathcal{S}$, who wishes to transmit a message M to a receiver, $\mathcal{R}$, using a number n of channels ("wires"), some of which are controlled by a malicious adversary $\mathcal{A}$. The goal is to send the message both *privately* and *reliably*. Since its introduction, SMT has been widely studied and optimized with respect to several different settings of parameters (for example, see [SA96, SNP04, ACH06, FFGV07, KS08]).

⋆ Supported in part by IBM Faculty Award, Xerox Innovation Group Award, the Okawa Foundation Award, Intel, Teradata, NSF grants 0716835, 0716389, 0830803, 0916574, BSF grant 2008411 and U.C. MICRO grant.

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 126–141, 2011.
© Springer-Verlag Berlin Heidelberg 2011

It was shown in the original paper that PSMT is possible if and only if the adversary "corrupts" a number of wires $t < \frac{n}{2}$.

The model of Secure Message Transmission *by Public Discussion* (SMT-PD) was formally introduced by Garay and Ostrovsky [GO08] as an important building block for achieving unconditionally secure multi-party computation (MPC) [BGW88, CCD88] on *sparse* (i.e., not fully connected) networks. (An equivalent setup was studied earlier in a different context by Franklin and Wright [FW98]—see Section 3.) In this model, in addition to the wires in the standard SMT formulation, called "common" or "private" wires from now on, $\mathcal{S}$ and $\mathcal{R}$ gain access to a *public* channel which the adversary can read but not alter. In this new setting, secure message transmission is achievable even if the adversary corrupts up to $t < n$ of the private wires—i.e., up to all but one.

The motivation for this abstraction comes from the feasibility in partially connected settings for a subset of the nodes in the network to realize a broadcast functionality [PSL80, LSP82] (aka the Byzantine Generals Problem) despite the limited connectivity [DPPU86, Upf92, BG93][1], which plays the role of the public channel. (The private wires would be the multiple paths between them; see Section 3 for a more detailed exposition of the motivating scenario.)

In this short survey paper, after formally defining the SMT-PD problem, we overview the basic constructions starting with the first, rather communication-inefficient solutions to the problem, and ending with the most efficient solutions known to-date—optimal private communication and sublinear public communication. As in the case of PSMT, SMT-PD protocols come with an associated round complexity, defined as the number of information flows, which can only occur in one direction at a time, between $\mathcal{S}$ and $\mathcal{R}$, or vice versa. However, in the case of SMT-PD, where two types of communication channels—private and public—exist, the round complexity must account for the use of both. We also review results on this measure. These complexities refer to resource use for a single execution of an SMT-PD protocol. We finally review the *amortized* complexity of the problem, which would arise in natural use-case scenarios where $\mathcal{S}$ and $\mathcal{R}$ must send several messages back and forth, where later messages depend on earlier ones.

The presentation in general is at a high level, with references to the original publications where the mentioned results appeared for further reading, except for the treatment of optimal private communication in Section 5, where we go into slightly more detail for a variety of reasons, including: (1) the protocol presented there makes explicit use of randomness extractors, which is instructive as randomness extractors are to be credited for the reduction in the amount of transmitted randomness, which in turn is reflected in the gain in private communication; (2) the protocol is used as a building block in the following section, to achieve SMT-PD with sublinear public communication. Some of the background material for this more detailed exposition—error-correcting codes and consistency checks for codewords, and randomness extractors—is presented in the Appendix.

Related problems. As mentioned above, the first variant of SMT considered in the literature is *perfectly secure message transmission* (PSMT), in which both privacy and

[1] Called "almost-everywhere" agreement, or broadcast, in this setting, since not all uncorrupted parties may agree on or output the broadcast value.

reliability are perfect [DDWY93]. It is shown in the original paper that PSMT is possible if and only if $n \geq 2t + 1$. For such n, 2 rounds are necessary and sufficient for PSMT, while one-round PSMT is possible if and only if $n \geq 3t + 1$.

The communication complexity of PSMT depends on the number of rounds. For 1-round PSMT, Fitzi *et al.* [FFGV07] show that transmission rate $\geq \frac{n}{n-3t}$ is necessary and sufficient. (Recall that $n > 3t$ is required in this case.) For 2-round PSMT, Srinathan *et al.* [SNP04] show that a transmission rate $\geq \frac{n}{n-2t}$ is required[2]; this was extended in [SPR07], which showed that increasing the number of rounds does not help. Kurosawa and Suzuki [KS08] construct the first efficient (i.e., polynomial-time) 2-round PSMT protocol which matches this optimal transmission rate.

A number of relaxations of the perfectness requirements of PSMT are considered in the literature to achieve various tradeoffs (see for example [CPRS08] for a detailed discussion of variants of SMT). The most general version of SMT (or SMT-PD) is perhaps (ϵ, δ)-SMT. We will call a protocol for SMT(-PD) an (ϵ, δ)-SMT(-PD) protocol provided that the adversary's advantage in distinguishing any two messages is at most ϵ, and the receiver correctly outputs the message with probability $1 - \delta$. The lower bound $n \geq 2t + 1$ holds even in this general setting (at least for non-trivial protocols, such as those satisfying $\epsilon + \delta < 1/2$); hence the most interesting case for SMT-PD is the case when the public channel is required: $t < n \leq 2t$.

For the "by Public Discussion" part of the name in the SMT-PD problem formulation, Garay and Ostrovsky drew inspiration from the seminal work on privacy amplification and secret-key agreement by Bennett *et al.* [BBR88, BBCM95] where two honest parties can also communicate through a public and authentic channel, as well as through a private channel which an adversary can partially eavesdrop or tamper. This problem has been studied extensively over the years under different variants of the original model. In a sense and at a high level, SMT-PD can be considered as a specialized instance of the original privacy amplification model, for a specific structure of the private communication, for example, viewing the communication over the multiple private wires between $\mathcal{S}$ and $\mathcal{R}$ as "blocks" over a single channel, and a specific adversarial tampering function, where one of the blocks is to remain private and unchanged.

2 Model and Problem Definition

Definition 1. *If X and Y are random variables over a discrete space S, the* statistical distance between X and Y *is defined to be*

$$\Delta(X, Y) \stackrel{\text{def}}{=} \frac{1}{2} \sum_{s \in S} \left| \Pr[X = s] - \Pr[Y = s] \right| .$$

We say that X and Y are ϵ-close *if $\Delta(X, Y) \leq \epsilon$.*

The *public discussion model* for secure message transmission [GO08] consists of a Sender $\mathcal{S}$ and Receiver $\mathcal{R}$ (PPTMs) connected by n communication channels, or *wires*, and one *public channel*. $\mathcal{S}$ wishes to send a message $M_{\mathcal{S}}$ from message space $\mathcal{M}$ to $\mathcal{R}$,

[2] The authors claim a matching upper bound as well, but this was shown to be flawed [ACH06].

and to this end $\mathcal{S}$ and $\mathcal{R}$ communicate with each other in synchronous rounds in which one player sends information across the wires and/or public channel. Communication on the public channel is reliable but public; the common wires may be corrupted and so are not necessarily reliable or private.

$\mathcal{A}$ is a computationally unbounded adversary who seeks to disrupt the communication and/or gain information on the message. $\mathcal{A}$ may *adaptively* corrupt up to $t < n$ of the common wires (potentially all but one!). Corrupted wires are actively controlled by $\mathcal{A}$: he can eavesdrop, block communication, or place forged messages on them. Further, we assume $\mathcal{A}$ is *rushing*—in each round, he observes what is sent on the public channel and all corrupted wires before deciding what to place on corrupted wires, or whether to corrupt additional wires (which he then sees immediately).

An *execution* E of an SMT-PD protocol is determined by the random coins of $\mathcal{S}$, $\mathcal{R}$, and $\mathcal{A}$ (which we denote $C_{\mathcal{S}}$, $C_{\mathcal{R}}$, $C_{\mathcal{A}}$ respectively), and the message $M_{\mathcal{S}} \in \mathcal{M}$. The *view of a player* $\mathcal{P} \in \{\mathcal{S}, \mathcal{R}, \mathcal{A}\}$ *in an execution* E, denoted $\mathrm{View}_{\mathcal{P}}$, is a random variable consisting of $\mathcal{P}$'s random coins and all messages received (or overheard) by $\mathcal{P}$. ($\mathcal{S}$'s view also includes $M_{\mathcal{S}}$). Additionally, let $\mathrm{View}_{\mathcal{P}}(M_0)$ denote the distribution on $\mathrm{View}_{\mathcal{P}}$ induced by fixing $M_{\mathcal{S}} = M_0$. In each execution, $\mathcal{R}$ outputs a received message $M_{\mathcal{R}}$, a function of $\mathrm{View}_{\mathcal{R}}$.

We can now define an (ϵ, δ)-SMT-PD protocol (cf. [FW98, GO08, SJST09]):

Definition 2. *A protocol* Π *in the model above, in which* $\mathcal{S}$ *attempts to send a message* $M_{\mathcal{S}}$ *to* $\mathcal{R}$*, is* (ϵ, δ)-secure *(or simply, is an* (ϵ, δ)-SMT-PD protocol*) if it satisfies:*

PRIVACY*: For any two messages* $M_0, M_1 \in \mathcal{M}$*,* $\mathrm{View}_{\mathcal{A}}(M_0)$ *and* $\mathrm{View}_{\mathcal{A}}(M_1)$ *are* ϵ*-close.*

RELIABILITY*: For all* $M_{\mathcal{S}} \in \mathcal{M}$ *and all adversaries* $\mathcal{A}$*,* $\mathcal{R}$ *should correctly receive the message with probability at least* $1 - \delta$*; i.e.,* $\Pr[M_{\mathcal{R}} = M_{\mathcal{S}}] \geq 1 - \delta$*. (The probability is taken over all players' random coins.)*

As in the case of PSMT, SMT-PD protocols come with an associated *round complexity*, defined as the number of information flows, which can only occur in one direction at a time, between $\mathcal{S}$ and $\mathcal{R}$, or vice versa. However, in the case of SMT-PD, where two types of communication channels—private and public—exist, the round complexity of a protocol is specified by the tuple (X, Y), meaning a total of X communication rounds, Y of which must use the public channel.

3 The First Solutions

As mentioned above, SMT-PD was introduced by Garay and Ostrovsky as an enabling building block for achieving MPC [BGW88, CCD88] on partially connected networks, a notion that they termed *almost-everywhere* MPC [GO08]. Recall that in the original MPC setting with unconditional security (i.e., no bounds are assumed on the computational power of the adversary), n parties are assumed to be interconnected by a complete graph of pairwise reliable and private channels. Such strong connectivity, however, is far from modeling the actual connectivity of real communication networks, which is what led researchers, starting with the seminal work of Dwork, Peleg, Pippenger and

Upfal [DPPU86], to study the achievability of distributed, fault-tolerant tasks such as Byzantine agreement [PSL80, LSP82] on networks with sparse connectivity, where not all parties share point-to-point reliable and private channels.

As pointed out in [DPPU86], an immediate observation about this setting is that one may not be able to guarantee agreement amongst all honest parties, and some of them, say x, must be given up, which is the reason for the "almost-everywhere" task qualifier. Thus, in this line of work the goal is to be able to tolerate a high value for t (a constant fraction of n is the best possible) while minimizing x.[3] Recall that Byzantine agreement and broadcast (aka the Byzantine Generals Problem), where there is only one sender and the rest of the parties are to agree on the sender's value, are two closely related tasks. The observation made by Garay and Ostrovsky was that "almost-everywhere" broadcast could readily instantiate a public channel between all pairs of nodes where the task was possible (specifically, amongst $(n-t-x)$-many nodes), while the multiple, potentially not disjoint paths would play the role of the private wires.

Going back to SMT-PD specifics, Garay and Ostrovsky first describe a (4,3)-round $(0, \delta)$ protocol which was subsequently improved by the authors to (3,2) rounds [Gar08]. The protocol has the following basic structure, which has been kept by most of subsequent work:

1. In the first round, one of the parties (in their case $\mathcal{R}$) sends lots of randomness on each private wire.
2. Using the public channel, $\mathcal{R}$ then sends checks to verify the randomness sent in the first round was not tampered with.
3. $\mathcal{S}$ discards any tampered wires, combines each remaining wire's randomness to get a one-time pad R, and sends $C = M \oplus R$ on the public channel, where M is the message to be sent.

We refer to [GO08] for details on the protocol. Now, although acceptable as a feasibility result, the protocol has linear transmission rate on both the public and private channels, which does sound excessive, as for example, given the amount of randomness that is needed to "blind" the message in the last round, a constant transmission rate on the public channel should in principle suffice. Indeed, reducing both public and private communication has been the goal of subsequent work, presented in the following sections.

Unnoticed by Garay and Ostrovsky at the time, however, was the fact that work done earlier by Franklin and Wright [FW98] in a slightly different message transmission context would yield an equivalent setup. Specifically, Franklin and Wright studied a model where $\mathcal{S}$ and $\mathcal{R}$ would be connected by n *lines*, each comprising a sequence of m nodes, not counting sender and receiver. In this model, they consider *multicast* as the only communication primitive. A message that is multicast by any node is (authentically, and only) received by all its neighbors—i.e., both neighbors of an "internal" node, or all n neighbors of $\mathcal{S}$ and $\mathcal{R}$.

[3] As shown in the original paper, the dependency on d, the degree of the network, to achieve this goal is paramount. See [CGO10] for the state of the art on efficient (i.e., polynomial-time) agreement and MPC protocols on small-degree networks.

They present protocols for reliable and secure communication for the multicast model, and, importantly, they show an equivalence between networks with multicast and those with simple lines and broadcast (i.e., the public discussion model). A first SMT-PD protocol results as a consequence of that equivalence. The resulting protocol also has round complexity $(3, 2)$ as the [GO08] protocol[4]; however, when $t < n < \lceil \frac{3t}{2} \rceil$ (including the worst case $t = n+1$), their protocol has (pick your poison) either positive privacy error $\epsilon > 0$, or *exponential* communication complexity. Refer to [FW98] for further details on the protocol.

In addition, Franklin and Wright show the following impossibility result (using our current terminology):

Theorem 3 ([FW98]). *Perfectly reliable ($\delta = 0$) SMT-PD protocols are impossible when $n \leq 2t$.*

On the other hand, perfect privacy ($\epsilon = 0$) is possible, and is achieved by the two protocols mentioned above, as well as by the more efficient ones reviewed in the sequel.

4 Round Complexity

The round complexity of both SMT-PD protocols mentioned in the previous section is $(3, 2)$, again meaning 3 total number of rounds, 2 of which use the public channel. However, it was not known whether this round complexity was optimal. In [SJST09], Shi, Jiang, Safavi-Naini and Tuhin show that this is indeed the case, namely, that the minimum values of X and Y for which an (X, Y)-round (ϵ, δ)-SMT-PD protocol can exist are 3 and 2, respectively. We now overview their approach (the following paragraph is taken from [SJST09] almost *verbatim*).

The result is obtained in three steps. First, they prove that there is no $(2, 2)$-round (ϵ, δ)-SMT-PD protocol with $\epsilon + \delta < 1 - \frac{1}{|\mathcal{M}|}$ when $n \leq 2t$, where $\mathcal{M}$ denotes the message space, meaning that such protocols with $(2, 2)$ round complexity will be either unreliable or insecure. In the second step they show that when the party, $\mathcal{S}$ or $\mathcal{R}$, who will invoke the public channel does not depend on the protocol execution but is statically determined by the protocol specification, then there is no $(X, 1)$-round (ϵ, δ)-SMT-PD protocol, $X \geq 3$, with $\epsilon + \delta < 1 - \frac{1}{|\mathcal{M}|}$ and $\delta < \frac{1}{2}(1 - \frac{1}{|\mathcal{M}|})$ when $n \leq 2t$. Lastly, they generalize this last step to the case where the invoker of the public channel is not fixed at the start of the protocol, but instead adaptively determined in each execution, and show that there is no $(3, 1)$-round (ϵ, δ)-SMT-PD protocol with $3\epsilon + 2\delta < 1 - \frac{3}{|\mathcal{M}|}$.

We remark that at a high level, the approach to proving these lower bounds is similar in spirit to that taken for the connectivity lower bound of $n \geq 2t + 1$ for PSMT [DDWY93]. Namely, it is assumed toward contradiction that a protocol with $n = 2t$ exists. Then, an adversary is considered who randomly corrupts either the first or the last t wires, and then follows the protocol specification to "impersonate" $\mathcal{S}$ and $\mathcal{R}$ to

[4] The round complexity is not apparent from the text, for two reasons: (1) The protocol is described in terms of the multicast model, not SMT-PD directly; and (2) the authors consider synchronous "rounds" not in the abstract SMT-PD model, but in the more concrete setting of nodes relaying messages in the underlying network.

each other on the corrupted wires. Formally, Shi *et al.* define a relation **W** on protocol executions, where $(E, E') \in \mathbf{W}$ (E and E' are called *swapped executions*) if the following holds:

In execution E:
- $\mathcal{S}$ has message $M_{\mathcal{S}}$ and coins $C_{\mathcal{S}}$;
- $\mathcal{R}$ has coins $C_{\mathcal{R}}$;
- $\mathcal{A}$ corrupts the *first* t wires, impersonates $\mathcal{S}$ using message $M_{\mathcal{A}}$ and coins $C_{\mathcal{AS}}$, and impersonates $\mathcal{R}$ using coins $C_{\mathcal{AR}}$.

In execution E':
- $\mathcal{S}$ has message $M_{\mathcal{A}}$ and coins $C_{\mathcal{AS}}$;
- $\mathcal{R}$ has coins $C_{\mathcal{AR}}$;
- $\mathcal{A}$ corrupts the *last* t wires, impersonates $\mathcal{S}$ using message $M_{\mathcal{S}}$ and coins $C_{\mathcal{S}}$, and impersonates $\mathcal{R}$ using coins $C_{\mathcal{R}}$.

When no public channel is available (as in [DDWY93]), $\mathcal{S}$ and $\mathcal{R}$ can simply never distinguish whether they are in E or E', rendering PSMT (indeed, any (ϵ, δ)-SMT for non-trivial parameter choices) impossible. When, as here, the public channel is available, $\mathcal{S}$ and $\mathcal{R}$ can leverage it to separate true messages from fakes, but only following sufficient interaction (hence the round lower bounds).

To give a better flavor for why "sufficient interaction" entails (3,2) round complexity, consider the second step described above, which appears as the following theorem:

Theorem 4 ([SJST09]). *Let $n \leq 2t$ and $X \geq 3$. Then an $(X, 1)$-round (ϵ, δ)-SMT-PD protocol with fixed invoker of public channel has either $\epsilon + \delta \geq 1 - \frac{1}{|\mathcal{M}|}$ or $\delta \geq \frac{1}{2}(1 - \frac{1}{|\mathcal{M}|})$.*

We now give a high-level sketch of the proof of this theorem. Though we omit full technical details (see [SJST09]), we hope to capture the essence of the argument.

Assume that Π is an (ϵ, δ)-SMT-PD protocol which invokes the public channel only once, with fixed invoker.

Case 1: $\mathcal{R}$ invokes the public channel. Reliability will be broken. Observe that prior to any invocation of the public channel, the parties are in the same situation as if no public channel existed, hence $\mathcal{A}$'s impersonations are entirely undetectable. During this portion of the execution, $\mathcal{S}$ cannot send more than ϵ information about the message on either the first t or the last t wires, at risk of violating ϵ-privacy.

At some point, $\mathcal{R}$ invokes the public channel. Now, $\mathcal{S}$ may detect which set of wires is corrupted. However, it is of no use: with the public channel no longer available, $\mathcal{S}$ has no way of reliably getting this knowledge to $\mathcal{R}$. Therefore $\mathcal{A}$, after viewing $\mathcal{R}$'s public message, can simply continue to impersonate $\mathcal{S}$ towards $\mathcal{R}$ as before (taking into account how the impersonation would respond to the public transmission). $\mathcal{R}$ will be unable to distinguish between two swapped executions E and E'. Any time he outputs the correct message in E, he outputs the incorrect message in E', and vice versa—the exception being the $1/|\mathcal{M}|$ mass of swapped executions where $M_{\mathcal{S}} = M_{\mathcal{A}}$. Hence $\mathcal{R}$ can do essentially no better than $1/2$ at correctly outputting $M_{\mathcal{S}}$, when $|\mathcal{M}|$ is large.

Case 2: $\mathcal{S}$ invokes the public channel. Either privacy or reliability is broken. As before, prior to invoking the public channel, $\mathcal{S}$ cannot send more than δ information about the message on either the first or the last t wires, and $\mathcal{S}$ and $\mathcal{R}$ cannot distinguish between a certain pair of swapped executions.

Eventually $\mathcal{S}$ invokes the public channel (on which he cannot send more than δ information without breaking privacy!). $\mathcal{R}$ may now be able to tell which set of wires is corrupted. From this point forward, $\mathcal{A}$ will impersonate $\mathcal{R}$ as though he received the public transmission sent by $\mathcal{S}$. Therefore $\mathcal{S}$ is still unable to distinguish between a certain swapped pair E and E'. Now $\mathcal{S}$ faces a dilemma: if he sends enough information on even one of the first or last t sets of wires to determine the message with high probability, then with probability $1/2$ privacy is broken; on the other hand, if he does not send this information, then reliability is broken.

For full details of the above argument, as well as the proof that $(2, 2)$-round SMT-PD is impossible and the extension to an adaptively chosen invoker, we refer the interested reader to [SJST09].

In addition to the lower bound, Shi *et al.* present a new (ϵ, δ)-SMT-PD protocol with *constant* transmission rate on the public channel, as opposed to linear as in [GO08], as well as linear transmission rate on the private channels. (See upcoming sections.)

5 SMT-PD with Optimal Private Communication

We start this section by stating the minimal private communication complexity required by any (ϵ, δ)-SMT-PD protocol, as shown by Garay, Givens and Ostrovsky [GGO10]:

Theorem 5 ([GGO10]). *Let Π be any (ϵ, δ)-SMT-PD protocol with $n \leq 2t$, in the presence of a passive, non-adaptive adversary $\mathcal{A}$. Let C denote the expected communication (in bits) over the private wires (the expectation is taken over all players' coins and the choice of $M_S \in \mathcal{M}$). Then*

$$C \geq \frac{n}{n-t} \cdot (-\log(1/|\mathcal{M}| + 2\epsilon) - H_2(\sqrt{\delta}) - 2\sqrt{\delta}\log|\mathcal{M}|),$$

where $H_2(\cdot)$ denotes the binary entropy function. In particular, if $\epsilon = O(1/|\mathcal{M}|)$ and $\delta = O(1)$, then $C = \Omega(mn/(n-t))$, where $m = |M_S|$.

We mentioned above the linear transmission rate in private communication incurred by protocols in [GO08, SJST09], essentially meaning an n-fold overhead for the transmission of a message, compared to the $\Omega(n/(n-t))$ bound above. We now reproduce a basic (ϵ, δ)-SMT-PD protocol presented in [GGO10] matching this bound. Here we present the "generic" version of the protocol; refer to [GGO10] for alternative instantiations.

In [GGO10] the protocol is called Π_{Gen} (for "generic"). Π_{Gen} relies on two primitives as black boxes: an error-correcting code $\mathcal{E}$ and an average-case strong extractor, Ext_A (see Appendix). The efficiency of the protocol depends on the interaction between the basic parameters of the protocol—ϵ, δ, m, n, and t—and the parameters of $\mathcal{E}$ and Ext_A. At a high level, the protocol has the same basic structure outlined in Section 3.

Protocol $\Pi_{\mathrm{Gen}}(\epsilon, \delta, m, n, t, \mathcal{E}, \mathrm{Ext}_A)$

1. ($\mathcal{R} \overset{PRI}{\rightarrow} \mathcal{S}$). For each wire i, $\mathcal{R}$ chooses a random $r_i \in \{0,1\}^K$ and sends the codeword $\mathcal{C}_i = Enc(r_i)$ along wire i. Let $\mathcal{C}_i^*$ be the codeword received by $\mathcal{S}$, and $r_i^* = Dec(\mathcal{C}_i^*)$.
2. ($\mathcal{R} \overset{PUB}{\rightarrow} \mathcal{S}$). $\mathcal{R}$ chooses a random subset $J = \{j_1, j_2, \ldots, j_\ell\} \subset [N]$ of codeword indices, $|J| = \ell$. Let
$$\mathcal{C}_i|_J = (\mathcal{C}_{i,j_1}, \mathcal{C}_{i,j_2}, \ldots, \mathcal{C}_{i,j_\ell}) \in \{0,1\}^\ell$$
be the codeword $\mathcal{C}_i$ restricted to the indices of J. $\mathcal{R}$ sends $(J, \{\mathcal{C}_i|_J\}_{i\in[n]})$ to $\mathcal{S}$ over the public channel.
3. ($\mathcal{S} \overset{PUB}{\rightarrow} \mathcal{R}$). $\mathcal{S}$ rejects any wire i which is syntactically incorrect (including the case that $\mathcal{C}_i^*$ is not a valid codeword), or for which $\mathcal{C}_i|_J$ conflicts with $\mathcal{C}_i^*$. Call the set of remaining, accepted wires ACC, and let $B \in \{0,1\}^n$, where $b_i = 1 \iff i \in \mathsf{ACC}$. Let α^* denote the concatenation of r_i^* for all $i \in \mathsf{ACC}$, padded with zeroes so that $|\alpha^*| = nK$. $\mathcal{S}$ chooses $seed \in \{0,1\}^s$ uniformly at random. He applies $\mathrm{Ext}_A : \{0,1\}^{nK} \times \{0,1\}^s \rightarrow \{0,1\}^m$ to obtain $R^* = \mathrm{Ext}_A(\alpha^*, seed)$, where $|R^*| = m$. $\mathcal{S}$ puts $C = M_{\mathcal{S}} \oplus R^*$, and sends $(B, C, seed)$ on the public channel.
Receiver: $\mathcal{R}$ uses B to reconstruct ACC. He forms α by concatenating r_i for each $i \in \mathsf{ACC}$, and padding with zeroes to size nK. He applies $\mathrm{Ext}_A : \{0,1\}^{nK} \times \{0,1\}^s \rightarrow \{0,1\}^m$, obtaining $R = \mathrm{Ext}_A(\alpha, seed)$. He then recovers $M_{\mathcal{R}} = C \oplus R$.

Fig. 1. A generic SMT-PD protocol with optimal communication complexity on the private wires and linear communication complexity on the public channel

However, the use of extractors allows to reduce the amount of transmitted randomness, which is reflected in the gain in private communication.

One remark is that one may modify Π_{Gen} to have interaction order $\mathcal{S}$-$\mathcal{R}$-$\mathcal{S}$, instead of $\mathcal{R}$-$\mathcal{R}$-$\mathcal{S}$ as presented here. One advantage of $\mathcal{R}$-$\mathcal{R}$-$\mathcal{S}$ is that when instantiated with deterministic extractors, it does not require any random coins for $\mathcal{S}$ (in contrast to $\mathcal{S}$-$\mathcal{R}$-$\mathcal{S}$, where both parties use randomness crucially).

Let error-correcting code $\mathcal{E}$ have encoding and decoding functions $Enc : \{0,1\}^K \rightarrow \{0,1\}^N$ and $Dec : \{0,1\}^N \rightarrow \{0,1\}^K$, respectively, and relative minimum distance D. (K is specified below.) While $N > K$ may be arbitrarily large for the purpose of correctness, K/N and D are both required to be constant for the complexity analysis—that is, $\mathcal{E}$ is *asymptotically good*.

Second, let Ext_A be an average-case $(nK, m, k_{min}, \epsilon/2)$-strong extractor. Here K is, as above, the source length of the error-correcting code $\mathcal{E}$, and m and ϵ are the message-length and privacy parameters of Π_{Gen}. k_{min} is the min-entropy threshold. Now clearly $m \leq k_{min} \leq nK$. On the other hand, it is required that $k_{min} = O(m)$ for the complexity claim to hold—that is, Ext_A should extract a constant fraction of the min-entropy. Further, the extractor's seed length s should be $O(n + m)$.

Finally, let $b = \frac{1}{1-D}$, and then set $\ell = \lceil \log_b(t/\delta) \rceil$. Now with foresight, set $K = \lceil k_{min}/(n-t) \rceil + \ell$.[5] Note that if $k_{min} = O(m)$, then $K = O(m)/(n-t) + \ell$. The protocol, Π_{Gen}, is presented in Fig. 1.

The following is shown in [GGO10]:

Theorem 6 ([GGO10]). *Let $t < n$. Protocol Π_{Gen} is a $(3,2)$-round (ϵ,δ)-SMT-PD protocol with communication complexity $O(\frac{mn}{n-t})$ on the private wires provided that $m/(n-t) = \Omega(\log(t/\delta))$, and communication complexity $\max(O(\log(t/\delta)(n+\log m)), O(m+n))$ on the public channel, provided only that $m = \Omega(\log(t/\delta))$.*

Refer to [GGO10] for details on the proof and complexity analysis of the above theorem, as well as for possible instantiations of Π_{Gen}. For example, for 0-private protocols, the most important instantiation would be that with Reed-Solomon codes and the extractor Ext_q of Appendix B. Nevertheless, other choices of explicit extractor, such as Kamp and Zuckerman's deterministic symbol-fixing extractor [KZ06], are possible.

6 Reducing Public Communication

Protocol Π_{Gen} from the previous section, while achieving optimal private communication, incurs a cost of size m on the public channel in its Round 3 communication. The same (i.e., linear public communication) holds for the SMT-PD protocol by Shi *et al.* [SJST09] alluded to in Section 4. However, as mentioned earlier, the *implementation* of a public channel on point-to-point networks is costly and highly non-trivial in terms of rounds of computation and communication, as already the sending of a single message to a node that is not directly connected is simulated by sending the message over multiple paths, not just blowing up the communication but also incurring a slowdown factor proportional to the diameter of the network, and this is a process that must be repeated many times—linear in the number of corruptions for deterministic, error-free broadcast protocols (e.g., [GM98]), or expected (but high) constant for randomized protocols [FM97, KK06]—which makes minimizing the use of this resource by SMT-PD protocols an intrinsically compelling issue.

We now overview a protocol for SMT-PD presented in [GGO10] which achieves logarithmic communication complexity (in m) *on the public channel*. In addition, the protocol is perfectly private, achieves the optimal communication complexity of $O(\frac{mn}{n-t})$ on the private wires, and has optimal round complexity of $(3,2)$.

The improvement comes from the insight that $\mathcal{S}$ can send the third-round message (C, in the notation of Π_{Gen}) on the *common* wires, provided that $\mathcal{S}$ *authenticates* the transmission (making use of the public channel). $\mathcal{S}$ could simply send C on every common wire and authenticate C publicly. The downside of this approach is that the private wire complexity would then be $\Omega(mn)$ rather than $O(\frac{mn}{n-t})$—no longer optimal. The solution presented in [GGO10] is to take C and encode it *once again* using Reed-Solomon codes into shares $C_1, \ldots, C_n$, each of size $\approx \frac{m}{n-t}$, such that any $n-t$ correct C_i's will reconstruct C. $\mathcal{S}$ then sends C_i on wire i, and authenticates each C_i publicly.

[5] As a sanity check, observe that $k_{min} \leq nK = n(k_{min}/(n-t) + \ell)$, so the extractor we define can exist.

This authentication uses a short secret key, call it s^*, of size $\ell(n + \log(\frac{cm}{n-t}))$ (which is the cost of authenticating n messages of size $cm/(n-t)$, using the consistency check of Appendix A; c is an absolute constant). Thus, $\mathcal{S}$ and $\mathcal{R}$ run *two processes in parallel*: a "small" strand, in which $\mathcal{S}$ privately sends the short key to $\mathcal{R}$; and a "big" strand, in which $\mathcal{S}$ sends $M_{\mathcal{S}}$ to $\mathcal{R}$, making use of the shared key in the third round. The small protocol sends the short key using any reasonably efficient SMT-PD protocol—for example, Π_{Gen} from Section 5, instantiated with Reed-Solomon codes. In order to achieve perfect privacy and optimal private wire complexity, Garay, Givens and Ostrovsky also use Π_{Gen} with Reed-Solomon codes for the big strand of the protocol. Call the resulting protocol Π_{SPD} (for "small" public discussion). As a result, they are able to show the following:

Theorem 7 ([GGO10]). *Protocol* Π_{SPD} *is a valid* $(3,2)$*-round* $(0,3\delta)$*-SMT-PD protocol. It has communication complexity* $O(\frac{mn}{n-t})$ *on the private wires and* $O(n\log(t/\delta)\log m)$ *on the public channel, provided* $m = \Omega(n\log(t/\delta)\log q)$.

7 Amortized SMT-PD

As mentioned in Section 1, the motivation behind the formulation of SMT-PD was for such a protocol to be used as a subroutine multiple times in a larger protocol, in which case a natural question is whether some of the lower bounds on resource use for a single execution of SMT-PD can be beaten *on average* through amortization. For instance, an almost-everywhere MPC protocol may invoke an SMT-PD subroutine every time any two nodes in the underlying network need to communicate. Must they use the public channel twice every single time, or can the nodes involved, say, save some state information which allows them to reduce their use of the public channel in later invocations?

In [GGO10], Garay, Givens and Ostrovsky show that amortization can in fact drastically reduce the use of the public channel: indeed, it is possible to limit the total number of uses of the public channel to *two*, no matter how many messages are ultimately sent between two nodes. (Since two uses of the public channel are required to send any reliable communication whatsoever, this is best possible.)

Of course, $\mathcal{S}$ and $\mathcal{R}$ may use the first execution of SMT-PD to establish a shared secret key, which can then be used for message encryption and authentication on the common wires. The Sender computes a ciphertext and sends it (with authentication) on every common wire. With overwhelming probability, no forged message is accepted as authentic, and the Receiver accepts the unique, authentic message which arrives on any good wire. However, since we are considering the information-theoretic setting, each use of the shared key reduces its entropy with respect to the adversary's view. If the parties know in advance an upper bound on the total communication they will require, and can afford to send a proportionally large shared key in the first execution of SMT-PD, then this approach is tenable by itself.

In some situations, however, the players may not know a strict upper bound on the number of messages they will send. And even when they do, it may happen that the protocol terminates early with some probability, so that an initial message with large entropy is mostly wasted. With these considerations in mind, it is worth exploring strategies which allow $\mathcal{S}$ and $\mathcal{R}$ to communicate *indefinitely* after using only two broadcast

rounds and a limited initial message. The approach in [GGO10] is to separate Sender and Receiver's interaction following the first execution of SMT-PD into two modes: a *Normal Mode* and a *Fault-Recovery Mode*.

In the Normal Mode, $\mathcal{S}$ and $\mathcal{R}$ communicate over the common wires without making use of their shared key; they are successful provided the adversary does not actively interfere. However, if the adversary does interfere, one of the players (say $\mathcal{R}$) will detect this and enter Fault-Recovery Mode, in which he uses the shared key to broadcast information about the messages he received on each common wire, allowing $\mathcal{S}$ to determine at least one corrupted wire (which he then informs $\mathcal{R}$ about, authentically).

In this way, $\mathcal{S}$ and $\mathcal{R}$ communicate reliably and privately so long as the adversary is passive; and any time he is active, they are able to eliminate at least one corrupted wire. This is achieved in [GGO10] by defining a weaker version of SMT-PD in which reliability is only guaranteed for a passive adversary—i.e., if the adversary only eavesdrops, then $\mathcal{R}$ receives the message correctly; however, if the adversary actively corrupts any wire, then with probability $\geq 1 - \delta$, *either* $\mathcal{R}$ receives the message correctly ($M_{\mathcal{R}} = M_{\mathcal{S}}$), *or* $\mathcal{R}$ outputs "Corruption detected." In [GGO10], the authors present a protocol that achieves Weak SMT-PD in one round.

Note that Weak SMT-PD, as sketched above, is similar in spirit to *almost* SMT from the standard (non-public discussion) model [KS07], in that both are relaxations which allow one-round transmission (for Weak SMT-PD, only with a passive adversary). The difference is that in the ordinary model, definitions for almost SMT require that the message be correctly received with overwhelming probability regardless of the adversary's actions; in the public discussion model, when the adversary controls a majority of wires, this is impossible, so it is only required that corruptions be detected. Indeed, one cannot guarantee reliability in a single round even when the adversary simply *blocks* transmission on corrupted wires (otherwise a minority of wires would carry enough information to recover the message, thus violating privacy).

Theorem 8 ([GGO10]). *Given an initial shared secret consisting of $O(n^2)$ field elements, $\mathcal{S}$ and $\mathcal{R}$ can communicate* indefinitely *using only the private wires. The probability that one of them will* ever *accept an incorrect message is $\leq t\delta$. Moreover, with probability $\geq 1 - t\delta$, $\mathcal{A}$ gains at most δ information on each of t different messages, and no information on any other message.*

8 Summary and Future Work

In this brief survey we have reviewed the motivation behind the formulation of the SMT-PD problem, as well as presented a historical overview of existing constructions, culminating with an SMT-PD protocol that achieves optimal private communication and sublinear public communication, in the optimal number of rounds [GGO10]. Specifically (and assuming for simplicity $\delta = O(1)$), the protocol has public channel communication complexity $O(n \log n \log m)$, where m is the size of the message, for messages of sufficient size, namely, $m/\log m = \Omega(n \log n)$. An immediate question is whether these bounds—public communication as well as messages sizes for which it can be achieved—can be improved.

References

[ACH06] Agarwal, S., Cramer, R., de Haan, R.: Asymptotically optimal two-round perfectly secure message transmission. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 394–408. Springer, Heidelberg (2006)

[BBCM95] Bennett, C.H., Brassard, G., Crèpeau, C., Maurer, U.: Generalized privacy amplification. IEEE Transactions on Information Theory 41(6), 1015–1923 (1995)

[BBR88] Bennett, C.H., Brassard, G., Robert, J.M.: Privacy amplification by public discussion. Siam Journal of Computing 17(2) (1988)

[BG93] Berman, P., Garay, J.: Fast consensus in networks of bounded degree. Distributed Computing 2(7), 62–73 (1991); Preliminary version in WDAG 1990

[BGW88] Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: STOC, pp. 1–10 (1988)

[CCD88] Chaum, D., Crepeau, C., Damgard, I.: Multiparty unconditionally secure protocols. In: STOC, pp. 11–19 (1988)

[CGO10] Chandran, N., Garay, J., Ostrovsky, R.: Improved fault tolerance and secure computation on sparse networks. In: Abramsky, S., Gavoille, C., Kirchner, C., Meyer auf der Heide, F., Spirakis, P.G. (eds.) ICALP 2010. LNCS, vol. 6199, pp. 249–260. Springer, Heidelberg (2010)

[CPRS08] Choudhary, A., Patra, A., Pandu Rangan, C., Srinathan, K.: Unconditionally reliable and secure message transmission in undirected synchronous networks: Possibility, feasibility and optimality. Cryptology ePrint Archive, Report 2008/141 (2008)

[DDWY93] Dolev, D., Dwork, C., Waarts, O., Yung, M.: Perfectly secure message transmission. Journal of ACM 1(40), 17–47 (1993)

[DORS08] Dodis, Y., Ostrovsky, R., Reyzin, L., Smith, A.: Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. SIAM J. Comput. (2008)

[DPPU86] Dwork, C., Peleg, D., Pippinger, N., Upfal, E.: Fault tolerance in networks of bounded degree. In: STOC, pp. 370–379 (1986)

[FFGV07] Fitzi, M., Franklin, M.K., Garay, J.A., Vardhan, S.H.: Towards optimal and efficient perfectly secure message transmission. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 311–322. Springer, Heidelberg (2007)

[FM97] Feldman, P., Micali, S.: An optimal probabilistic protocol for synchronous Byzantine agreement. SIAM J. Comput. 26(4), 873–933 (1997)

[FW98] Franklin, M., Wright, R.: Secure communication in minimal connectivity models. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 346–360. Springer, Heidelberg (1998)

[Gar08] Garay, J.A.: Partially connected networks: Information theoretically secure protocols and open problems (Invited talk). In: Safavi-Naini, R. (ed.) ICITS 2008. LNCS, vol. 5155, p. 1. Springer, Heidelberg (2008)

[GGO10] Garay, J., Givens, C., Ostrovsky, R.: Secure message transmission with small public discussion. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 177–196. Springer, Heidelberg (2010); Full version in Cryptology ePrint Archive, Report 2009/519

[GM98] Garay, J., Moses, Y.: Fully polynomial Byzantine agreement for $n > 3t$ processors in $t + 1$ rounds. SIAM J. Comput. 27(1), 247–290 (1998); Prelim. in STOC 1992

[GO08] Garay, J.A., Ostrovsky, R.: Almost-everywhere secure computation. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 307–323. Springer, Heidelberg (2008)

[KK06] Katz, J., Koo, C.-Y.: On expected constant-round protocols for byzantine agreement. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 445–462. Springer, Heidelberg (2006)

[KS07] Kurosawa, K., Suzuki, K.: Almost secure (1-round, n-channel) message transmission scheme. Cryptology ePrint Archive, Report 2007/076 (2007)
[KS08] Kurosawa, K., Suzuki, K.: Truly efficient 2-round perfectly secure message transmission scheme. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 324–340. Springer, Heidelberg (2008)
[KZ06] Kamp, J., Zuckerman, D.: Deterministic extractors for bit-fixing sources and exposure-resilient cryptography. SIAM J. Comput. 36(5), 1231–1247 (2006)
[LSP82] Lamport, L., Shostak, R., Pease, M.: The Byzantine generals problem. ACM Transactions on Programming Languages and Systems, 382–401 (July 1982)
[MS83] Macwilliams, F., Sloane, N.: The Theory of Error-Correcting Codes. North-Holland, Amsterdam (1983)
[PSL80] Pease, M., Shostak, R., Lamport, L.: Reaching agreement in the presence of faults. Journal of the ACM, JACM 27(2) (April 1980)
[SA96] Sayeed, H., Abu-Amara, H.: Efficient perfectly secure message transmission in synchronous networks. Information and Computation 1(126), 53–61 (1996)
[SJST09] Shi, H., Jiang, S., Safavi-Naini, R., Tuhin, M.: Optimal secure message transmission by public discussion. In: IEEE Symposium on Information Theory (2009)
[SNP04] Srinathan, K., Narayanan, A., Pandu Rangan, C.: Optimal perfectly secure message transmission. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 545–561. Springer, Heidelberg (2004)
[SPR07] Srinathan, K., Prasad, N.R., Pandu Rangan, C.: On the optimal communication complexity of multiphase protocols for perfect communication. In: IEEE Symposium on Security and Privacy, pp. 311–320 (2007)
[Upf92] Upfal, E.: Tolerating linear number of faults in networks of bounded degree. In: PODC, pp. 83–89 (1992)

A Error-Correcting Codes and Consistency Checks for Codewords

For the results alluded to in this paper, the following definition of error-correcting codes is sufficient:

Definition 9. *Given a finite alphabet Σ, an* error-correcting code $\mathcal{E}$ of minimum distance d *is a pair of mappings $Enc : \Sigma^K \to \Sigma^N$, where $K < N$ and $Dec : \Sigma^N \to \Sigma^K$, such that (1) any two distinct elements x, y in the image of Enc (the* codewords*) have $dist(x, y) \geq d$ in the Hamming metric; (2) $Dec(Enc(x)) = x$ for all $x \in \Sigma^K$.*[6] *We say $\mathcal{E}$ has* rate K/N *and* relative minimum distance d/N.

The protocols presented here require a family of codes of increasing input length which is *asymptotically good*, that is, $\mathcal{E}$ should have *constant* rate and *constant* relative minimum distance D. See, e.g., [MS83] for a standard reference.

Of particular interest are the well-known Reed-Solomon codes over F_q, obtained by oversampling polynomials in $\mathbb{F}_q[X]$. Given an input in $\mathbb{F}_q^K$, we interpret it as a polynomial f of degree $\leq K - 1$; to obtain a codeword from f, we simply evaluate it at N distinct points in $\mathbb{F}_q$, for any $N > K$. Indeed, any two such polynomials agree on at most $K-1$ points, therefore the Reed-Solomon code has minimum distance $N-K+1$.

[6] Note in particular that this allows us to test for membership in the image $Enc(\Sigma^K)$ by first decoding and then re-encoding.

Protocols make use of a simple method to probabilistically detect when codewords sent on the private wires are altered by $\mathcal{A}$. Simply put, the sender of the codeword reveals a small subset of the codeword symbols. Formally, suppose $\mathcal{S}$ sends a codeword $\mathcal{C} \in \Sigma^N$ to $\mathcal{R}$ over one of the private wires, and $\mathcal{R}$ receives the (possibly altered) codeword $\mathcal{C}^*$. (If $\mathcal{R}$ receives a non-codeword, he immediately rejects it.) Then to perform the consistency check, $\mathcal{S}$ chooses a random set $J = \{j_1, j_2, \ldots, j_\ell\} \subset [N]$ and sends $(J, \mathcal{C}|_J)$ to $\mathcal{R}$, where $\mathcal{C}|_J$ represents the codeword $\mathcal{C}$ restricted to the indices in J. If the revealed symbols match, then the consistency check succeeds; otherwise the check fails and $\mathcal{R}$ rejects $\mathcal{C}^*$ as tampered.

Suppose $\mathcal{A}$ alters $\mathcal{C}$ to a different codeword, $\mathcal{C}^* \neq \mathcal{C}$. Since $\mathcal{C}$ and $\mathcal{C}^*$ are distinct valid codewords, they differ in at least, say, 1/3 of their symbols. Therefore, the probability that they agree on a randomly chosen index is $\leq 2/3$, and so

$$\Pr[\mathcal{R} \text{ accepts } \mathcal{C}^*] = \Pr[\mathcal{C}|_J = \mathcal{C}^*|_J] \leq (2/3)^\ell.$$

Thus, with probability $\geq 1 - (2/3)^\ell$, $\mathcal{R}$ will reject a tampered codeword. Of course, the validity of the check depends upon $\mathcal{A}$ not knowing J at the time of potential corruption of $\mathcal{C}$.

B Average Min-Entropy and Average-Case Randomness Extractors

Recall that the *min-entropy* of a distribution $X = (X_1, \ldots, X_N)$ over $\{0,1\}^N$ is defined as

$$H_\infty(X) = \min_x \left(-\log\left(\Pr[X = x]\right)\right),$$

and gives a measure of the amount of randomness "contained" in a weakly random source. We say a distribution X is a k_{min}-*source* if $H_\infty(X) \geq k_{min}$.

A *(seeded)* $(N, M, k_{min}, \epsilon)$-*strong extractor* is a (deterministic) function

$$\text{Ext} : \{0,1\}^N \times \{0,1\}^D \to \{0,1\}^M$$

such that for *any* k_{min}-source X, the distribution $U_D \circ \text{Ext}(X, U_D)$ is ϵ-close to $U_D \circ U_M$ (where U_k represents the uniform distribution on $\{0,1\}^k$). The input to the extractor is the N-bit k_{min}-source, X, together with a truly random seed s, which is uniformly distributed over $\{0,1\}^D$. Its output is an M-bit string which is statistically close to uniform, *even conditioned on the seed s used to generate it.*

This notion of min-entropy, and of a general randomness extractor, may be an awkward fit when considering an adversary with side information Y as above. In these cases, a more appropriate measure may be found in the *average min-entropy* of X given Y, defined in [DORS08] by

$$\tilde{H}_\infty(X \mid Y) = -\log\left(\mathbb{E}_{y \leftarrow Y}\left[\max_x \Pr\left[X = x \mid Y = y\right]\right]\right).$$

Note that this definition is based on the *worst-case* probability for X, conditioned on the *average distribution* (as opposed to worst-case probability) of Y. The rationale is

that Y is assumed to be outside of the adversary's control; however, once Y is known, the adversary then predicts the *most likely* X, given that particular Y.

[DORS08] use average min-entropy to define an object closely related to extractors: A *(seeded) average-case* $(N, M, k_{min}, \epsilon)$*-strong extractor* is a (deterministic) function

$$\mathrm{Ext} : \{0,1\}^N \times \{0,1\}^D \to \{0,1\}^M$$

such that the distribution of $(U_D \circ \mathrm{Ext}(X, U_D), I)$ is ϵ-close to $(U_D \circ U_M, I)$, whenever (X, I) is a jointly distributed pair satisfying $\tilde{H}_\infty(X \mid I) \geq k_{min}$. The similarity to an ordinary extractor is clear. [DORS08] prove the following fact about average min-entropy:

Fact 10. *If Y has at most 2^ℓ possible values, then $\tilde{H}_\infty(X \mid (Y, Z)) \geq \tilde{H}_\infty(X \mid Z) - \ell$.*

Extracting randomness from $\mathbb{F}_q$. Some of the instantiations in this paper make use of a special-purpose *deterministic* (seedless) extractor Ext_q which operates at the level of field elements in $\mathbb{F}_q$ as opposed to bits. Ext_q works not on general min-entropy sources, but on the restricted class of *symbol-fixing sources*, which are strings in $\mathbb{F}_q^N$ such that some subset of K symbols is distributed independently and uniformly over $\mathbb{F}_q$, while the remaining $N - K$ symbols are fixed. Given a sample from any such source, Ext_q outputs K field elements which are uniformly distributed over $\mathbb{F}_q^K$.

Ext_q works as follows: Given $\alpha \in \mathbb{F}_q^N$, construct $f \in \mathbb{F}_q[X]$ of degree $\leq N - 1$, such that $f(i) = \alpha_i$ for $i = 0, \ldots, N - 1$. Then $\mathrm{Ext}_q(\alpha) = (f(N), f(N + 1), \ldots, f(N + K - 1))$. (Of course we require $N + K \leq q$.) This extractor has proven useful in previous SMT protocols as well (see, e.g., [ACH06, KS08]).

Variations on Encoding Circuits for Stabilizer Quantum Codes

Markus Grassl

Centre for Quantum Technologies, National University of Singapore,
S15 #03-11, 3 Science Drive 2, Singapore 117543, Republic of Singapore
Markus.Grassl@nus.edu.sg

Abstract. Quantum error-correcting codes (QECC) are an important component of any future quantum computing device. After a brief introduction to stabilizer quantum codes, we present two methods to efficiently compute encoding circuits for them.

1 Introduction

The development of quantum error-correcting codes (QECC) has been an important step towards building information processing devices that exploit the many aspects of quantum mechanics. In this article, we give a brief introduction to a large class of QECCs, the so-called stabilizer (quantum) codes. They are closely related to classical error-correcting codes. Stabilizer codes can also be efficiently described in terms of a so-called stabilizer matrix which corresponds to the generator matrix of classical linear codes. We will show how a specific standard form of the stabilizer matrix can be used to derive efficient encoding circuits for stabilizer codes. The first method is a variation of a method developed by Cleve and Gottesman [3] for the simplest case of so-called qubits. The second method uses a connection to graph codes introduced by Schlingemann and Werner [15]. Both methods are illustrated for an optimal single-error correcting qutrit code $\mathcal{C} = [\![6, 2, 3]\!]_3$.

2 Stabilizer Quantum Codes

In this section we give a brief introduction to stabilizer quantum codes and their relation to classical codes. In order to fix notation, we start with the mathematical model of quantum mechanical systems. Further information can, for example, be found in the book by Nielsen and Chuang [13].

2.1 Qudit Systems

A pure state of a quantum mechanical system can be described by a normalized vector in a complex Hilbert space $\mathcal{H}$. In our context, the dimension of the Hilbert space is always finite. In particular, we consider the situation when $\dim \mathcal{H} = q = p^m$, p prime, is a prime power. Then we can label the elements of an orthonormal

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 142–158, 2011.
© Springer-Verlag Berlin Heidelberg 2011

basis of $\mathcal{H}$ with the elements of a finite field $\mathbb{F}_q$. In the following, we will use Dirac's bra-ket notation. A (column) vector $(x_1, \ldots, x_d)^T \in \mathcal{H} = \mathbb{C}^d$ will be denoted by a so-called ket $|x\rangle$. An element y of the dual space $\mathcal{H}^*$ corresponds to a row vector $(\overline{y}_1, \ldots, \overline{y}_d)$ and is denoted by a so-called bra $\langle y|$, where $\overline{y}_i$ denotes complex conjugation.

With this preparation, we can define a qudit system as follows:

Definition 1 (single qudit system). *A qudit system is a quantum mechanical system whose state space is a complex Hilbert space $\mathcal{H}$ of finite dimension d. If $d = q = p^m$ is a prime power, an orthonormal basis of $\mathcal{H}$ is given by*

$$\mathcal{B} = \{|\alpha\rangle \colon \alpha \in \mathbb{F}_q\}\,.$$

A pure state of the qudit system is any normalized vector in $\mathcal{H}$ which can be expressed by

$$|\psi\rangle = \sum_{\alpha\in\mathbb{F}_q} c_\alpha |\alpha\rangle, \qquad \textit{where} \sum_{\alpha\in\mathbb{F}_q} |c_\alpha|^2 = 1.$$

Normalized vectors $|\psi\rangle$ and $|\psi'\rangle$ that are related by a complex number of modulus one (a so-called phase*), i. e., $|\psi'\rangle = \exp(i\theta)|\psi\rangle$, describe the same quantum state.*

The name *qudit* is derived from the short form *qubit* for a *quantum bit*, the special case of a two-dimensional complex Hilbert space.

Recall that a bra $\langle\beta|$ denotes an element of the dual space $\mathcal{H}^*$. The action of $\langle\beta|$ on the ket $|\alpha\rangle$ yields the Hermitian inner product of the corresponding vectors $|\alpha\rangle$ and $|\beta\rangle$, in particular $\langle\beta|\alpha\rangle = \delta_{\alpha,\beta}$ if $|\alpha\rangle$ and $|\beta\rangle$ are orthonormal basis states. Linear operators on $\mathcal{H}$ can be expressed as

$$M = \sum_{\alpha,\beta\in\mathbb{F}_q} \mu_{\alpha,\beta} |\alpha\rangle\langle\beta|.$$

In our context, one may think of $\mathcal{H}$ as the group algebra $\mathbb{C}[\mathbb{F}_q]$. The addition in $\mathbb{F}_q$ gives rise to linear operations on $\mathcal{H}$ defined as

$$X^\alpha = \sum_{\gamma\in\mathbb{F}_q} |\gamma + \alpha\rangle\langle\gamma|$$

with the composition rule $X^{\alpha_1} X^{\alpha_2} = X^{\alpha_1+\alpha_2}$. Using the additive characters $\chi_\beta(x) = \omega_p^{\mathrm{Tr}(\beta x)}$ of $\mathbb{F}_q$, where $\omega_p = \exp(2\pi i/p)$ is a primitive complex p-th root of unity and $\mathrm{Tr}\colon \mathbb{F}_q \to \mathbb{F}_p$ denotes the trace, we can define the following operators:

$$Z^\beta = \sum_{\gamma\in\mathbb{F}_q} \chi_\beta(\gamma) |\gamma\rangle\langle\gamma| = \sum_{\gamma\in\mathbb{F}_q} \omega_p^{\mathrm{Tr}(\beta\gamma)} |\gamma\rangle\langle\gamma|.$$

The operators Z^β obey a similar composition rule $Z^{\beta_1}Z^{\beta_2} = Z^{\beta_1+\beta_2}$ as the operators X^α. Furthermore, we compute

$$Z^\beta X^\alpha = \left(\sum_{\gamma\in\mathbb{F}_q} \omega_p^{\mathrm{Tr}(\beta\gamma)}|\gamma\rangle\langle\gamma|\right)\left(\sum_{\gamma'\in\mathbb{F}_q} |\alpha+\gamma'\rangle\langle\gamma'|\right) = \sum_{\gamma\in\mathbb{F}_q} \omega_p^{\mathrm{Tr}(\beta(\alpha+\gamma))}|\alpha+\gamma\rangle\langle\gamma|$$

$$= \omega_p^{\mathrm{Tr}(\beta\alpha)} \sum_{\gamma\in\mathbb{F}_q} \omega_p^{\mathrm{Tr}(\beta\gamma)}|\alpha+\gamma\rangle\langle\gamma| \qquad = \omega_p^{\mathrm{Tr}(\beta\alpha)} X^\alpha Z^\beta.$$

This implies

$$X^\alpha Z^\beta X^{\alpha'} Z^{\beta'} = \omega_p^{\mathrm{Tr}(\alpha'\beta)} X^\alpha X^{\alpha'} Z^\beta Z^{\beta'} \qquad = \omega_p^{\mathrm{Tr}(\alpha'\beta)} X^{\alpha'} X^\alpha Z^{\beta'} Z^\beta$$
$$= \omega_p^{\mathrm{Tr}(\alpha'\beta-\alpha\beta')} X^{\alpha'} Z^{\beta'} X^\alpha Z^\beta.$$

From these commutation relations it follows that the operators X^α and Z^β of order p generate a finite group.

Proposition 1 (one-qudit Pauli group). *The group*

$$\mathcal{P} = \left\langle X^\alpha, Z^\beta \colon \alpha,\beta\in\mathbb{F}_q\right\rangle = \left\{\omega_p^\gamma X^\alpha Z^\beta \colon \alpha,\beta\in\mathbb{F}_q, \gamma\in\mathbb{F}_p\right\}$$

generated by X^α and Z^β is a group of order pq^2. It is called the (generalized) one-qudit Pauli group, since for $p = 2$, the matrices X and Z correspond to the Pauli matrices $\sigma_x = \left(\begin{smallmatrix}0&1\\1&0\end{smallmatrix}\right)$ and $\sigma_z = \left(\begin{smallmatrix}1&0\\0&-1\end{smallmatrix}\right)$, respectively.

In the context of quantum error-correcting codes, it is important to note that the q^2 matrices $\{X^\alpha Z^\beta \colon \alpha,\beta\in\mathbb{F}_q\}$ form a basis of the vector space of complex $q\times q$ matrices.

Combining two quantum systems which are described by Hilbert spaces $\mathcal{H}_A$ and $\mathcal{H}_B$, the states of the joint system are in the Hilbert space $\mathcal{H}_{AB} = \mathcal{H}_A\otimes\mathcal{H}_B$. Note that the dimension of the joint space is the product of the dimensions of its components. Combining n systems of equal dimension, the dimension of the joint space grows exponentially with n.

Definition 2 (n-qudit system). *The combined Hilbert space of n qudit systems with corresponding Hilbert spaces of equal dimension d is the n-fold tensor product of the individual Hilbert spaces which is isomorphic to a complex vector space of dimension d^n. For $d = q = p^m$ a prime power, we can label the elements of an orthonormal basis $\mathcal{B}$ by vectors in $\mathbb{F}_q^n$, i. e.,*

$$\mathcal{B} = \{|\boldsymbol{x}\rangle \colon \boldsymbol{x}\in\mathbb{F}_q^n\},$$

where we identify

$$|x_1\rangle\otimes|x_2\rangle\otimes\cdots\otimes|x_n\rangle = |x_1\rangle|x_2\rangle\ldots|x_n\rangle = |x_1,x_2,\ldots,x_n\rangle = |\boldsymbol{x}\rangle.$$

A pure state of an n-qudit system is any normalized vector which can be expressed by

$$|\psi\rangle = \sum_{\boldsymbol{x}\in\mathbb{F}_q^n} c_{\boldsymbol{x}}|\boldsymbol{x}\rangle, \qquad \textit{where} \sum_{\boldsymbol{x}\in\mathbb{F}_q^n} |c_{\boldsymbol{x}}|^2 = 1.$$

The single-qudit operators X and Z can be naturally generalized to operators on $(\mathbb{C}^q)^{\otimes n}$:

$$X^{\boldsymbol{\alpha}} = X^{\alpha_1} \otimes \cdots \otimes X^{\alpha_n} = \sum_{\boldsymbol{\gamma} \in \mathbb{F}_q^n} |\boldsymbol{\alpha} + \boldsymbol{\gamma}\rangle\langle\boldsymbol{\gamma}| \quad \text{and} \tag{1a}$$

$$Z^{\boldsymbol{\beta}} = Z^{\beta_1} \otimes \cdots \otimes Z^{\beta_n} = \sum_{\boldsymbol{\gamma} \in \mathbb{F}_q^n} \omega_p^{\mathrm{Tr}(\boldsymbol{\beta}\cdot\boldsymbol{\gamma})} |\boldsymbol{\gamma}\rangle\langle\boldsymbol{\gamma}|, \tag{1b}$$

where $\boldsymbol{\beta}\cdot\boldsymbol{\gamma} = \sum_{i=1}^n \beta_i\gamma_i$ is the standard (Euclidean) inner product of the vectors $\boldsymbol{\beta}, \boldsymbol{\gamma} \in \mathbb{F}_q^n$.

Proposition 2 (n-qudit Pauli group). *The n-qudit Pauli group $\mathcal{P}_n$ is a finite group of order pq^{2n} given by*

$$\mathcal{P}_n = \left\langle X^{\boldsymbol{\alpha}}, Z^{\boldsymbol{\beta}} \colon \boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{F}_q^n \right\rangle = \left\{ \omega_p^\gamma X^{\boldsymbol{\alpha}} Z^{\boldsymbol{\beta}} \colon \boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{F}_q^n, \gamma \in \mathbb{F}_p \right\}.$$

The commutation relation of two elements is given by

$$X^{\boldsymbol{\alpha}} Z^{\boldsymbol{\beta}} X^{\boldsymbol{\alpha}'} Z^{\boldsymbol{\beta}'} = \omega_p^{\mathrm{Tr}(\boldsymbol{\alpha}'\cdot\boldsymbol{\beta} - \boldsymbol{\alpha}\cdot\boldsymbol{\beta}')} X^{\boldsymbol{\alpha}'} Z^{\boldsymbol{\beta}'} X^{\boldsymbol{\alpha}} Z^{\boldsymbol{\beta}}. \tag{2}$$

As a preparation for quantum error-correcting codes, we define a weight function on the Pauli group $\mathcal{P}_n$.

Definition 3. *The weight of $\omega_p^\gamma X^{\boldsymbol{\alpha}} Z^{\boldsymbol{\beta}} = \omega_p^\gamma (X^{\alpha_1} Z^{\beta_1}) \otimes \cdots \otimes (X^{\alpha_n} Z^{\beta_n})$ is the number of positions i where the tensor factor $X^{\alpha_i} Z^{\beta_i}$ is different from a scalar multiple of identity, i. e., $(\alpha_i, \beta_i) \neq (0, 0)$.*

2.2 Clifford Operations

For the quantum circuits to encode stabilizer code we will use some additional quantum operations. It turns out that conjugation by any of those operations maps any n-qudit Pauli matrix to a possibly different n-qudit Pauli matrix (see also [9]).

The first operation corresponds to multiplication by an element $\mu \in \mathbb{F}_q \setminus \{0\}$ and is defined as

$$M_\mu = \sum_{\gamma \in \mathbb{F}_q} |\mu\gamma\rangle\langle\gamma|.$$

The second operation is the discrete Fourier transformation for the additive group of $\mathbb{F}_q$

$$F = \frac{1}{\sqrt{q}} \sum_{\alpha, \beta \in \mathbb{F}_q} \omega_p^{\mathrm{Tr}(\alpha\beta)} |\alpha\rangle\langle\beta|.$$

Proposition 3 (see [9]). *The matrices M_μ and F act on the single-qudit Pauli group as follows:*

$$M_\mu^{-1} X^\alpha Z^\beta M_\mu = X^{\mu^{-1}\alpha} Z^{\mu\beta},$$
$$F^{-1} X^\alpha Z^\beta F = Z^{-\alpha} X^\beta.$$

The definition of the third type of operations is slightly more involved. For q odd we have

$$P_\mu = \sum_{\alpha\in\mathbb{F}_q} \omega_p^{-\operatorname{Tr}(\frac{1}{2}\mu\alpha^2)} |\alpha\rangle\langle\alpha|.$$

For $q = 2^m$, let $B = \{b_1, \ldots, b_m\}$ be an arbitrary (trace) self-dual basis of $\mathbb{F}_{2^m}$ over $\mathbb{F}_2$. We define a weight function on $\mathbb{F}_q$ as

$$\operatorname{wgt}\colon \mathbb{F}_q \to \mathbb{Z},\ \alpha \mapsto |\{j\colon j \in \{1,\ldots,m\}|\operatorname{Tr}(\alpha b_j) \neq 0\}|.$$

Then

$$P_{\mu^2} = M_\mu^{-1} \sum_{\alpha\in\mathbb{F}_q} i^{\operatorname{wgt}(\alpha)} |\alpha\rangle\langle\alpha| M_\mu, \qquad \text{where } i = \sqrt{-1} \in \mathbb{C}.$$

Note that for q even, every element in $\mathbb{F}_q$ is a square.

Proposition 4 (see [9]). *For every prime power q and $\mu \in \mathbb{F}_q \setminus \{0\}$, the operator P_μ acts on the single-qudit Pauli group as follows:*

$$P_\mu^{-1} X^\alpha Z^\beta P_\mu = \exp(i\theta) X^\alpha Z^{\mu\alpha+\beta}.$$

(For simplicity, we omit the details about the phase factor $\exp(i\theta)$ which can be found in [9]).

Finally, we will need some so-called controlled operations acting on two qudit systems.

Definition 4 (controlled Pauli gate). *For any generalized single-qudit Pauli matrix $M = X^\alpha Z^\beta$, a controlled-M gate with the first qudit as control and the second qudit as target is given by*

$$\textit{controlled-}M = \sum_{\gamma\in\mathbb{F}_q} |\gamma\rangle\langle\gamma| \otimes M^\gamma. \tag{3}$$

*Special cases are the controlled-*NOT *gate*

$$\textit{controlled-}X = \mathrm{CNOT} = \sum_{\gamma\in\mathbb{F}_q} |\gamma\rangle\langle\gamma| \otimes X^\gamma = \sum_{\alpha,\beta\in\mathbb{F}_q} |\alpha\rangle\langle\alpha| \otimes |\alpha+\beta\rangle\langle\beta|$$

mapping $|\alpha\rangle|\beta\rangle$ to $|\alpha\rangle|\alpha+\beta\rangle$, and the controlled-phase gate

$$\textit{controlled-}Z = \sum_{\gamma\in\mathbb{F}_q} |\gamma\rangle\langle\gamma| \otimes Z^\gamma = \sum_{\alpha,\beta\in\mathbb{F}_q} \omega_p^{\operatorname{Tr}(\alpha\beta)} |\alpha\rangle\langle\alpha| \otimes |\beta\rangle\langle\beta|,$$

which is diagonal and the role of control and target is symmetric.

Note that this definition can be extended to any $(n-1)$-qudit Pauli matrix $M = X^{\boldsymbol{\alpha}} Z^{\boldsymbol{\beta}}$, resulting in a controlled-M operation on n qudits.

2.3 Stabilizer Codes and Classical Codes

The theory of stabilizer quantum codes has been developed by Gottesman [4,5] and Calderbank et al. [2] who in particular established the relation to classical codes. The generalization to higher dimensional quantum systems has been put forward in [1].

Before introducing the basic concepts of stabilizer quantum codes, we consider the general situation. A quantum error-correcting code $\mathcal{C}$ of length n is a K-dimensional subspace of an n-fold tensor product of quantum systems of dimension q. We use the notation $\mathcal{C} = (\!(n, K)\!)_q$. The code is said to have minimum distance d, denoted as $\mathcal{C} = (\!(n, K, d)\!)_q$, if it

1. can correct arbitrary errors acting on strictly less than $d/2$ of the n subsystems, or
2. can correct arbitrary errors on no more than $d-1$ subsystems, where the position of the errors are known (erasures), or
3. can detect arbitrary errors on no more than $d-1$ subsystems or these errors have no effect on the code.

An error of weight w is any linear operator acting non-trivially on w of the n subsystems. As the elements of the n-qudit Pauli group of weight at most w form a basis of the vector space of such operators and by the fact that quantum error-correction is linear in the error operators, it is sufficient to deal with a finite set of errors, despite the fact that errors in quantum mechanics may depend on continuous parameters. For more details see, e. g., [11]. Note that for quantum codes, a non-trivial error can act trivially on the code if the code lies in an eigenspace of that operator. This fact is addressed by the second part of the third of the equivalent conditions.

This observation is closely related to the concept of stabilizer quantum codes.

Definition 5 (stabilizer quantum code). *Let $\mathcal{S}$ be an Abelian subgroup of the n-qudit Pauli group $\mathcal{P}_n$ that does not contain a non-trivial scalar multiple of identity. The stabilizer code with stabilizer $\mathcal{S}$ is the maximal linear subspace $\mathcal{C}$ stabilized by all elements of $\mathcal{S}$, i. e.,*

$$\mathcal{C} = \left\{ |\psi\rangle \in (\mathbb{C}^q)^{\otimes n} \mid \forall g \in \mathcal{S} \colon g|\psi\rangle = |\psi\rangle \right\}.$$

At the same time, $\mathcal{S}$ is the maximal subgroup of $\mathcal{P}_n$ that stabilizes $\mathcal{C}$, i. e.,

$$\mathcal{S} = \{ g \in \mathcal{P}_n \mid \forall |\psi\rangle \in \mathcal{C} \colon g|\psi\rangle = |\psi\rangle \}.$$

The dimension of the stabilizer code $\mathcal{C}$ and the size of its stabilizer $\mathcal{S}$ are related as follows:

Proposition 5. *If the stabilizer $\mathcal{S}$ of a stabilizer code $\mathcal{C}$ has order $p^{m(n-k)}$, the dimension of $\mathcal{C}$ is $p^{mk} = q^k$ and we use the notation $\mathcal{C} = [\![n, k]\!]_q$. Note that k need not be integral, but an integer multiple of $1/m$ if $|\mathcal{S}|$ is not a power of $q = p^m$.*

In order to establish a connection to classical error-correcting codes, first note that the elements of the n-qudit Pauli operators are—up to a multiplicative factor ω_p^γ referred to as *phase*—given by a pair of vectors $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{F}_q^n$ (see (1)). Since all elements of a stabilizer group $\mathcal{S}$ commute, we can ignore these phase factors and represent all elements of $\mathcal{S}$ just by vectors $(\boldsymbol{\alpha}|\boldsymbol{\beta}) \in \mathbb{F}_q^{2n}$ of length $2n$. Hence we get a mapping from $\mathcal{S}$ to an additively closed subset of $\mathbb{F}_q^{2n}$, i. e., an $\mathbb{F}_p$ vector space.

Definition 6 (stabilizer matrix, see also [9]). *Let $q = p^m$ and let $\mathcal{S}$ be an Abelian subgroup of $\mathcal{P}_n$ which does not contain a non-trivial multiple of identity. Furthermore, let $\{g_1, g_2, \ldots, g_\ell\}$ where $g_i = \omega^{\gamma_i} X^{\boldsymbol{\alpha}_i} Z^{\boldsymbol{\beta}_i}$ with $\gamma_i \in \mathbb{F}_p$ and $(\boldsymbol{\alpha}_i|\boldsymbol{\beta}_i) \in \mathbb{F}_q^{2n}$ be a minimal set of generators for $\mathcal{S}$. Then a stabilizer matrix of the corresponding stabilizer code $\mathcal{C}$ is a generator matrix of the (classical) additive code $C \subseteq \mathbb{F}_q^{2n}$ generated by $(\boldsymbol{\alpha}_i|\boldsymbol{\beta}_i)$. We will write this matrix in the form*

$$\left(\begin{array}{c|c} \boldsymbol{\alpha}_1 & \boldsymbol{\beta}_1 \\ \boldsymbol{\alpha}_2 & \boldsymbol{\beta}_2 \\ \vdots & \vdots \\ \boldsymbol{\alpha}_\ell & \boldsymbol{\beta}_\ell \end{array}\right) \in \mathbb{F}_q^{\ell \times 2n}.$$

The first n and the last n columns of this matrix are referred to as the X-part and the Z-part, respectively. If C is not only additively closed, but an $\mathbb{F}_q$-linear code with q^{n-k} codewords, we will pick a generator matrix of C with $n-k$ rows as stabilizer matrix. (Note that in that situation, for $q = p^m$ the stabilizer $\mathcal{S}$ has $m(n-k)$ generators.)

The commutation relation (2) allows us to characterize those additive codes $C \subseteq \mathbb{F}_q^{2n}$ that correspond to Abelian subgroups of $\mathcal{P}_n$.

Proposition 6. *An additive code $C \subseteq \mathbb{F}_q^{2n}$ corresponds to a stabilizer code $\mathcal{C}$ with stabilizer group $\mathcal{S}$ if and only if the code is self-orthogonal with respect to the trace symplectic inner product*

$$(\boldsymbol{\alpha}|\boldsymbol{\beta}) * (\boldsymbol{\alpha'}|\boldsymbol{\beta'}) = \mathrm{Tr}(\boldsymbol{\alpha'} \cdot \boldsymbol{\beta} - \boldsymbol{\alpha} \cdot \boldsymbol{\beta'}). \tag{4}$$

In other words, the code C is contained in its dual C^* with respect to (4)

$$C^* = \{(\boldsymbol{\alpha}|\boldsymbol{\beta}) \in \mathbb{F}_q^{2n} \mid \forall (\boldsymbol{\alpha'}|\boldsymbol{\beta'}) \in C \colon (\boldsymbol{\alpha}|\boldsymbol{\beta}) * (\boldsymbol{\alpha'}|\boldsymbol{\beta'}) = 0\}.$$

Note that if $q = p^m$, $m > 1$, the trace in (4) is quite often omitted, yielding the stronger condition $\boldsymbol{\alpha'} \cdot \boldsymbol{\beta} - \boldsymbol{\alpha} \cdot \boldsymbol{\beta'} = 0$.

The dual code C^* does have an interpretation in terms of operators on the complex vector space as well. It corresponds to the so-called *normalizer* of the quantum code $\mathcal{S}$ which is defined as

$$\mathcal{N}(\mathcal{S}) = \{h \in \mathcal{P}_n \mid \forall g \in \mathcal{S} \colon hg = gh\}.$$

(Strictly speaking, $\mathcal{N}$ is the centralizer of $\mathcal{S}$ in $\mathcal{P}_n$.) Since the elements of $\mathcal{N}$ commute with all elements of the stabilizer $\mathcal{S}$, they preserve the eigenspaces of $\mathcal{S}$, but act non-trivially on them. Hence in terms of error correction, the elements of $\mathcal{N} \setminus \mathcal{S}$ correspond to errors that cannot be detected, but have a non-trivial effect.

Proposition 7. *The minimum distance of a stabilizer code $\mathcal{C} = [\![n, k, d]\!]_q$ with stabilizer $\mathcal{S}$ and normalizer $\mathcal{N}$ is given by*

$$d = \min\{\operatorname{wgt} g \colon g \in \mathcal{N} \setminus \mathcal{S}\} \geq \min\{\operatorname{wgt} g \colon g \in \mathcal{N} \mid g \neq I\}.$$

At the same time, the operators in the normalizer $\mathcal{N}$ give rise to so-called *logical* or *encoded* operators $\overline{X}$ and $\overline{Z}$ acting on the code $\mathcal{C}$ and obeying the same commutation relations as the original operators X and Z. Again, these operators can be defined in terms of the classical additive code $C^* \subseteq \mathbb{F}_q^{2n}$.

Definition 7 (normalizer matrix). *Let $q = p^m$ and let $C \subseteq \mathbb{F}_q^{2n}$ be a classical additive code corresponding to a quantum code $\mathcal{C} = [\![n, k]\!]_q$ with stabilizer $\mathcal{S}$. Furthermore, let D be a self-dual code with $C \leq D = D^* \leq C^*$. The normalizer matrix of the quantum code $\mathcal{C}$ is a generator matrix of the classical code C^* of the form*

$$\left(\begin{array}{c|c} \boldsymbol{\alpha}_1 & \boldsymbol{\beta}_1 \\ \vdots & \vdots \\ \boldsymbol{\alpha}_{(n-k)m} & \boldsymbol{\beta}_{(n-k)m} \\ \hline \boldsymbol{\alpha}_1^Z & \boldsymbol{\beta}_1^Z \\ \vdots & \vdots \\ \boldsymbol{\alpha}_{mk}^Z & \boldsymbol{\beta}_{mk}^Z \\ \hline \boldsymbol{\alpha}_1^X & \boldsymbol{\beta}_1^X \\ \vdots & \vdots \\ \boldsymbol{\alpha}_{mk}^X & \boldsymbol{\beta}_{mk}^X \end{array}\right) \in \mathbb{F}_q^{((n+k)m)\times(2n)}.$$

The matrix is chosen such that (i) the first $(n-k)m$ rows generate C, (ii) the first nm rows generate D, (iii) all rows together generate C^*, and (iv) for $i \neq j$ the generators $(\boldsymbol{\alpha}_i^X|\boldsymbol{\beta}_i^X)$ and $(\boldsymbol{\alpha}_j^Z|\boldsymbol{\beta}_j^Z)$ are mutually orthogonal with respect to the trace symplectic inner product (4).

Again, if the code C is $\mathbb{F}_q$-linear, we will only list generators for the corresponding linear codes.

The generators $(\boldsymbol{\alpha}_i^X|\boldsymbol{\beta}_i^X)$ and $(\boldsymbol{\alpha}_j^Z|\boldsymbol{\beta}_j^Z)$ correspond to the aforementioned logical operators $\overline{X}$ and $\overline{Z}$, respectively.

The self-dual code D corresponds to a maximal Abelian subgroup of $\mathcal{P}_n$ that does not contain a non-trivial multiple of identity. The corresponding quantum code is one-dimensional, i. e., it contains only a single quantum state $|\overline{0}\rangle$.

Definition 8 (canonical basis). *A stabilizer code $\mathcal{C} = [\![n, k]\!]_{p^m}$ with stabilizer group $\mathcal{S}$ and logical operators $\overline{X}_i$ and $\overline{Z}_j$ has a basis given by*

$$|\overline{e_1, \ldots, e_{mk}}\rangle = \overline{X}_1^{e_1} \ldots \overline{X}_{mk}^{e_{mk}} |\overline{0}\rangle,$$

where $e_\ell \in \mathbb{F}_p$, and $|\overline{0}\rangle$ is the unique quantum state stabilized by all elements $g \in \mathcal{S}$ and all logical operators $\overline{Z}_j$.

3 Encoding Circuits from a Standard Form

We are now ready to present a method to derive an encoding circuit for a stabilizer quantum code from a standard form of the stabilizer matrix. This idea has been presented in [3] for qubit codes. (Note that there is a subtle error in the presentation in both [3] and the thesis [5], see [6].) The derivation presented below is similar to the algorithm discussed for qubits in [7].

Here we mainly focus on stabilizer codes for which the corresponding classical code is $\mathbb{F}_q$-linear. In the more general case, when the code is only $\mathbb{F}_p$-linear, one may consider the code $\mathcal{C} = [\![n, k, d]\!]_q$ as a code of $\mathcal{C}' = [\![nm, km, \geq d]\!]_p$, where $q = p^m$.

3.1 Standard Form of Stabilizers

First recall from Definition 5 that for a stabilizer code $\mathcal{C}$ with stabilizer $\mathcal{S}$, $g|\psi\rangle = |\psi\rangle$ for any state $|\psi\rangle \in \mathcal{C}$. Moreover, the operator

$$P_{\mathcal{C}} = \frac{1}{|\mathcal{S}|} \sum_{g \in \mathcal{S}} g \tag{5}$$

is a an orthogonal projection operator whose image is the code $\mathcal{C}$. If the Abelian group $\mathcal{S}$ is generated by $g_1, \ldots, g_\ell$, we can rewrite (5) as

$$P_{\mathcal{C}} = \frac{1}{|\mathcal{S}|} \prod_{i=1}^{\ell} \left(\sum_{j=0}^{p-1} g_i^j \right). \tag{6}$$

The following lemma allows us to replace the summation in (6) by a unitary operation.

Lemma 1. *Let g_i be a tensor product of n generalized Pauli matrices for qupit systems ($q = p$ prime) of the form $g_i = X \otimes \tilde{g}_i$. Furthermore, let*

$$P_{g_i} = \frac{1}{p} \sum_{j=0}^{p-1} g_i^j. \tag{7}$$

The action of P_{g_i} on a state of the form $|\psi\rangle = |0\rangle|\tilde{\psi}\rangle$ is, up to normalization, the same as that of a Fourier transformation on the first qupit followed by a controlled-$\tilde{g}_i$ operation on the remaining qupits.

Proof. Direct computation shows

$$\begin{aligned} P_{g_i}|\psi\rangle = \frac{1}{p}\sum_{j=0}^{p-1} g_i^j|\psi\rangle = \frac{1}{p}\sum_{j=0}^{p-1} X^j|0\rangle\otimes\tilde{g}_i^j|\tilde{\psi}\rangle = \frac{1}{p}\sum_{j=0}^{p-1}|j\rangle\otimes\tilde{g}_i^j|\tilde{\psi}\rangle \\ = \text{controlled-}\tilde{g}_i\left(\frac{1}{p}\sum_{j=0}^{p-1}|j\rangle\otimes|\tilde{\psi}\rangle\right) \\ = (\text{controlled-}\tilde{g}_i)(F\otimes I)\left(\frac{1}{\sqrt{p}}|0\rangle\otimes|\tilde{\psi}\rangle\right). \end{aligned}$$

□

Note that for prime powers $q = p^m$, $m > 1$, Lemma 1 still holds if the corresponding classical code is $\mathbb{F}_q$-linear, i.e., closed with respect to multiplication by elements of $\mathbb{F}_q$ (see in particular the definition (3) of a general controlled-gate).

This lemma allows us to implement the projection operator (5) by unitary operations provided that each of the generators g_i contains an operator X at a suitable position such that the operator X acts on a state $|0\rangle$. As the generators g_i mutually commute, we can apply them in any order.

Theorem 1 (stabilizer matrix in standard form). *For any stabilizer code $\mathcal{C} = [\![n, k, d]\!]_q$ corresponding to an $\mathbb{F}_q$-linear code, there exists a tensor product $T = t_1 \otimes \cdots \otimes t_n$ of single-qudit Clifford operations t_i and a permutation σ of the qudits such that the stabilizer matrix of the transformed code $\mathcal{C}' = \sigma T\mathcal{C}$ has the form*

$$\left(\begin{array}{cc|cc} I & B & C & D \end{array}\right), \tag{8}$$

where the diagonal of the square matrix C is zero.

Proof. Assume that the stabilizer matrix of $\mathcal{C}$ is of the form $(G^X|G^Z)$ with $n-k$ rows generating an $\mathbb{F}_q$-linear code. Linear operations on the rows of the stabilizer matrix do not change the corresponding stabilizer group. Hence, using Gaussian elimination (including simultaneous permutation of the columns in the X- and the Z-part), we can bring the X-part of the stabilizer matrix into the form

$$\left(\begin{array}{cc|cc} I_{r_1} & B' & C' & D' \\ 0 & 0 & E' & F' \end{array}\right),$$

where I_{r_1} is an $r_1 \times r_1$ identity matrix and r_1 is the rank of the X-part. Then, performing Gaussian elimination on F' we get

$$\left(\begin{array}{ccc|ccc} I_{r_1} & B_1' & B_2' & C'' & D_1'' & D_2'' \\ 0 & 0 & 0 & E_1'' & I_{r_2} & F_2'' \\ 0 & 0 & 0 & E_2'' & 0 & 0 \end{array}\right). \tag{9}$$

As the generators corresponding to the last group of rows commute with those corresponding to the first group, it follows that $E_2'' = 0$, i.e., the matrix F' has

full rank $r_2 = n - k - r_1$. Applying an inverse Fourier transformation to qudits $r_1 + 1$ to $r_1 + r_2$ we essentially swap the X- and the Z-part of those qudits and obtain

$$\left(\begin{array}{ccc|ccc} I_{r_1} & D_1'' & B_2' & C'' & -B_1' & D_2'' \\ 0 & I_{r_2} & E_1'' & 0 & 0 & F_2'' \end{array}\right).$$

Using row operations, we obtain the form (8). Finally, using the operation P_μ from Proposition 3, we can add a multiple of a column of the X-part to the corresponding column in the Z-part, thereby clearing the diagonal of C. □

Note that the standard form defined e.g. in [13] is essentially (9). That form can also be used to derive an encoding circuit. The following presentation, however, is slightly simplified by choosing the form (8).

If the stabilizer (matrix) of a quantum code is in the standard form of Theorem 1, we can use the technique of Lemma 1 to map any n-qudit state of the form $|\underbrace{0\ldots0}_{n-k}\rangle|\phi\rangle$, where $|\phi\rangle$ is an k-qudit state, to a state in the code.

A normalizer matrix of a quantum code which has a stabilizer matrix in standard form is given by

$$\left(\begin{array}{cc|cc} I_{n-k} & B & C & D \\ \hline 0 & I_k & D^T & 0 \\ \hline 0 & 0 & -B^T & I_k \end{array}\right), \tag{10}$$

where the first $n - k$ rows correspond to the stabilizer matrix in standard form (8) and next groups with k rows each correspond to the encoded X- and Z operators, respectively.

3.2 Example

We illustrate the construction of an encoding circuit for an optimal single-error correcting quantum code $\mathcal{C} = [\![6, 2, 3]\!]_3$ with the following stabilizer matrix:

$$(X|Z) = \left(\begin{array}{cccccc|cccccc} 1 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & 1 & 2 & 1 & 1 \\ 0 & 0 & 1 & 2 & 1 & 1 & 1 & 0 & 0 & 2 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 2 \\ 0 & 0 & 1 & 2 & 2 & 2 & 0 & 1 & 1 & 2 & 2 & 2 \end{array}\right). \tag{11}$$

This is a so-called quantum MDS code [10] as it meets the quantum version of the Singleton bound $2d + k \leq n + 2$ with equality. After Gaussian elimination (without permuting the columns) and applying suitable transformations P_μ to qudits $1, 3, 5$, we get a stabilizer matrix in standard form:

$$(X'|Z') = \left(\begin{array}{cccccc|cccccc} 1 & 0 & 0 & 2 & 0 & 2 & 0 & 2 & 0 & 1 & 1 & 2 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 2 \\ 0 & 0 & 1 & 2 & 0 & 0 & 2 & 2 & 0 & 2 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 2 & 1 & 1 & 0 & 0 & 2 \end{array}\right), \tag{12}$$

where the submatrices B and D are indicated in italics. From (10), the normalizer matrix in standard form reads

$$\left(\begin{array}{cccccc|cccccc} 1&0&0&\mathit{2}&0&\mathit{2}&0&2&0&\mathit{1}&1&\mathit{2}\\ 0&1&0&\mathit{0}&0&\mathit{0}&0&0&1&\mathit{2}&2&\mathit{2}\\ 0&0&1&\mathit{2}&0&\mathit{0}&2&2&0&\mathit{2}&0&\mathit{1}\\ 0&0&0&\mathit{0}&1&\mathit{1}&2&1&1&\mathit{0}&0&\mathit{2}\\ \hline 0&0&0&1&0&0&1&2&2&0&0&0\\ 0&0&0&0&0&1&2&2&1&0&2&0\\ \hline 0&0&0&0&0&0&1&0&1&1&0&0\\ 0&0&0&0&0&0&1&0&0&0&2&1 \end{array}\right). \tag{13}$$

The corresponding encoding circuit is shown in Fig. 1. For simplicity, we show an encoder for the code with stabilizer matrix (12). In order to obtain an encoder for the code with stabilizer matrix (11), the gates P_μ should be applied at the very end of the circuit. Each horizontal line corresponds to one qudit. A box on a line indicates the application of a Clifford operation to that qudit. Vertical lines indicate controlled Pauli operations, where the control-qudit is marked by a dot. Note that some simplifications are possible as gates Z acting on states $|0\rangle$ have no effect (see Fig. 2).

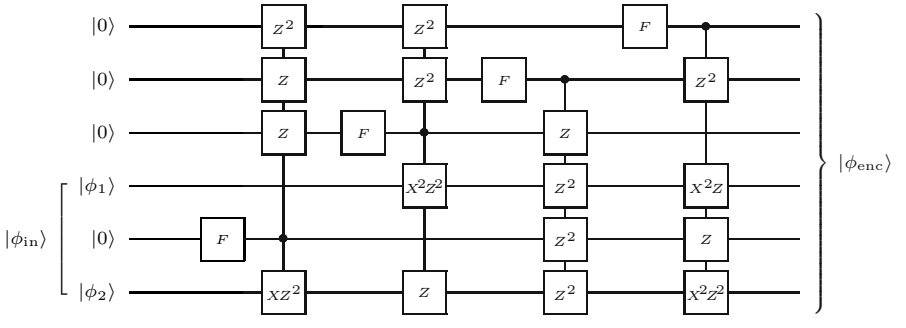


Fig. 1. Encoding circuit for a ternary quantum code $\mathcal{C} = [\![6, 2, 3]\!]_3$

4 Encoding Circuits Related to Graphs

The encoding circuits presented in this section are related to graphs. The concept of so-called graph codes was introduced by Schlingemann and Werner [15]. Later it was shown that every stabilizer quantum code can be represented as graph code and vice versa [8,14].

4.1 Stabilizer Codes and Graphs

Recall from Definition 8 that a canonical basis of a stabilizer code $\mathcal{C}$ is given by the action of the logical operators $\overline{X}_j$ on the unique quantum state $|\overline{0}\rangle$ stabilized

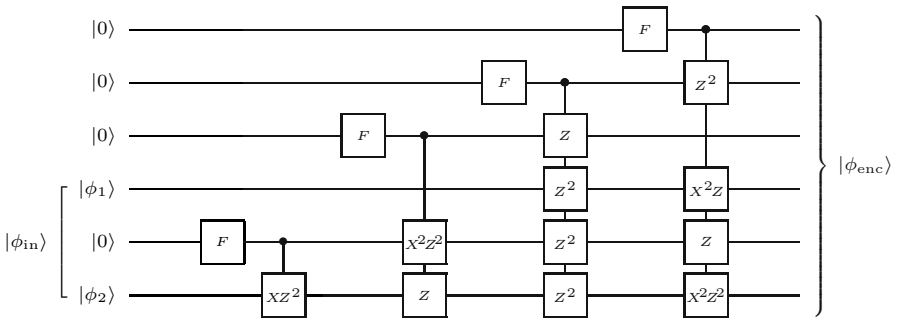


Fig. 2. Simplified encoding circuit for a ternary quantum code $\mathcal{C} = [\![6, 2, 3]\!]_3$. The controlled-$Z$ gates in Fig. 1 acting on the state $|0\rangle$ (those with the control "below" the target) have no effect and can be omitted.

by the original stabilizer group $\mathcal{S}$ and all logical operators $\overline{Z}_j$. The state $|\overline{0}\rangle$ can also be considered as a stabilizer code $\mathcal{C}_0$ of (complex) dimension one which is stabilized by the Abelian group $\mathcal{S}_0 = \langle \mathcal{S}, \overline{Z}_1, \ldots, \overline{Z}_{km} \rangle$. If we interchange the role of the logical operators $\overline{X}_j$ and $\overline{Z}_j$, a standard form of the stabilizer matrix for $\mathcal{S}_0$ can be easily computed from the normalizer matrix (10) of the code $\mathcal{C}$. We assume that the corresponding classical code is $\mathbb{F}_q$-linear, i.e, $\mathcal{S}$ is represented by a stabilizer matrix with $n - k$ rows. Using only row operations on the first n rows, the X-part can be transformed into an identity matrix. Using appropriate transformations P_μ, the diagonal of the Z-part can be set to zero. The stabilizer matrix for $\mathcal{S}_0$, together with the transformed X-operators, reads

$$\left(\begin{array}{c|c} I_n & A \\ \hline 0 & B \end{array}\right). \tag{14}$$

As the generalized Pauli matrices corresponding to the first n rows mutually commute, it follows that $A - A^T = 0$, i.e., the matrix A is symmetric. By construction, the diagonal of A is zero. Therefore, the matrix A can be used to define an undirected simple graph with n vertices and edges (i, j) labeled with the entries $A_{i,j}$.

Using Lemma 1, we obtain a quantum circuit for the preparation of the state $|\overline{0}\rangle$, starting with the unencoded state $|0 \ldots 0\rangle$. Note that the n-qudit Pauli matrices corresponding to the generators g_i of $\mathcal{S}_0$ given by (14) have exactly one tensor factor X at position i. The tensor factor at position j is $Z^{A_{ij}}$. The resulting controlled-Pauli operation consists only of controlled-Z operations for which control and target can be interchanged. The resulting quantum circuit has exactly the structure of the graph with the adjacency matrix A (see the left part of Fig. 3).

While in the previous section we have used only n qudits for the encoding circuit, we will now use k input qudits and n output qudits. For the encoding, we use the following transformation:

$$\left(\sum_{i_i,\ldots,i_k} \alpha_{i_1,\ldots,i_k}|i_1\ldots i_k\rangle\right) \otimes |\underbrace{\overline{0\ldots0}}_{n}\rangle \mapsto \sum_{i_i,\ldots,i_k} \alpha_{i_1,\ldots,i_k}|i_1\ldots i_k\rangle \otimes \left(\overline{X}_1^{i_1}\cdots\overline{X}_k^{i_k}|\overline{0\ldots0}\rangle\right). \tag{15}$$

From (14) it follows that the logical operators $\overline{X}_i$ are tensor products of powers of Z and identity. Hence the transformation (15) can be implemented using only controlled-Z as well. For the input qudit i as control, the targets and the powers of the Z-operation are determined by row i of the matrix B (see the middle part of Fig. 3).

At first glance, a problem with this approach is that the quantum information is contained both unencoded in the first k qudits of (15) and in encoded form in the last n qudits of (15). We cannot just discard the first k qudits, since that would destroy the encoded quantum information as well. Applying an inverse Fourier transformation to the first k qudits, we obtain (up to normalization) the state

$$\begin{aligned}&\sum_{j_i,\ldots,j_k} |j_1\ldots j_k\rangle \sum_{i_i,\ldots,i_k} \alpha_{i_1,\ldots,i_k}\omega^{-i_ij_i}\cdots\omega^{-i_kj_k}\left(\overline{X}_1^{i_1}\cdots\overline{X}_k^{i_k}|\overline{0\ldots0}\rangle\right)\\ =&\sum_{j_i,\ldots,j_k} |j_1\ldots j_k\rangle \sum_{i_i,\ldots,i_k} \alpha_{i_1,\ldots,i_k}\overline{Z}_1^{-j_1}\cdots\overline{Z}_k^{-j_k}\left(\overline{X}_1^{i_1}\cdots\overline{X}_k^{i_k}|\overline{0\ldots0}\rangle\right).\end{aligned} \tag{16}$$

The phase factors $\omega^{-i_\ell j_\ell}$ can be compensated for by controlled-$\overline{Z}_\ell$ operations yielding the state

$$\left(\sum_{j_i,\ldots,j_k} |j_1\ldots j_k\rangle\right) \otimes \sum_{i_i,\ldots,i_k} \alpha_{i_1,\ldots,i_k}\left(\overline{X}_1^{i_1}\cdots\overline{X}_k^{i_k}|\overline{0\ldots0}\rangle\right) = |\psi_0\rangle\otimes|\phi\rangle_{\text{enc}}.$$

This state is a tensor product of a superposition $|\psi_0\rangle$ of all basis states and the encoded state $|\phi\rangle_{\text{enc}}$. Therefore, we can discard the first k qudits without influencing the encoded state.

Instead of the controlled-$\overline{Z}_\ell$ operations, one can measure the first k qudits of the state after the inverse Fourier transformation (16) in the standard basis. This randomly projects the state onto one of the basis states $|j_1\ldots j_k\rangle$ and reveals the classical information $j_1,\ldots,j_k$. This information can then be used to apply the corresponding powers $\overline{Z}_\ell^{j_\ell}$ of the encoded operators. Note that if we do not apply the operators $\overline{Z}_\ell$, we still get a state in the underlying quantum code, but the very encoding depends on the random, but known measurement results.

4.2 Example

We illustrate the encoding circuit related to graphs using the same example as before. Starting with (13), the stabilizer matrix for $\mathcal{S}_0$ in standard form together with the encoded X-operators is given by

$$\left(\begin{array}{cccccc|cccccc} 1&0&0&0&0&0&0&0&0&1&0&2\\ 0&1&0&0&0&0&0&0&1&2&2&2\\ 0&0&1&0&0&0&0&1&0&2&0&1\\ 0&0&0&1&0&0&1&2&2&0&0&0\\ 0&0&0&0&1&0&0&2&0&0&0&2\\ 0&0&0&0&0&1&2&2&1&0&2&0\\ \hline 0&0&0&0&0&0&1&0&1&1&0&0\\ 0&0&0&0&0&0&1&0&0&0&2&1 \end{array}\right). \tag{17}$$

The encoded X- and Z operators from (13) remain unchanged, but their role is interchanged. The new operators are $\overline{X}_1 = ZIZZII$, $\overline{X}_2 = ZIIIZ^2Z$, $\overline{Z}_1 = ZZ^2Z^2XII$, and $\overline{Z}_2 = Z^2Z^2ZIIX$. The resulting encoding circuit using $2+6$ qutrits is shown in Fig. 3. The controlled-Z operators are depicted by a vertical line and crosses at both the control and the target. Doubling the vertical line indicates that the controlled-Z gate has to be applied twice, corresponding to an edge of weight two in the graph with adjacency matrix A.

Apart from the controlled-$\overline{Z}$ operators which have to be applied to disentangle the input qutrits from the output qutrits, the quantum circuit uses only $n+k$ Fourier operations and $O(n^2+kn)$ controlled-Z operations.

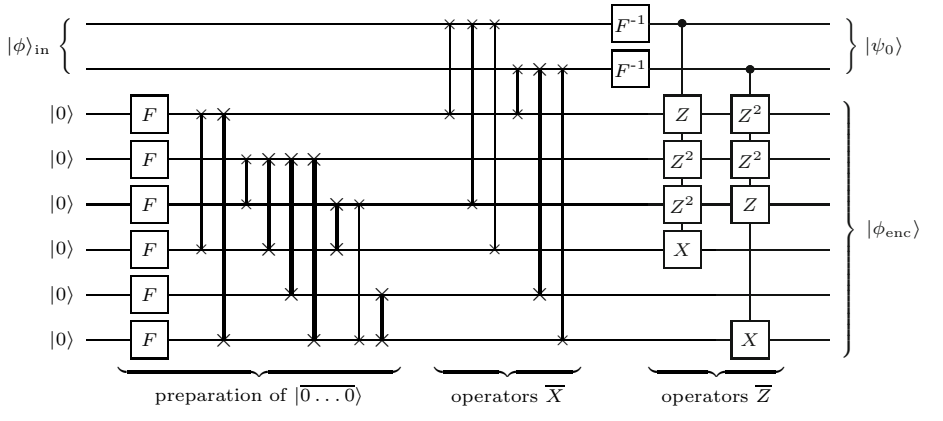


Fig. 3. Encoder for a ternary quantum code $\mathcal{C} = [\![6, 2, 3]\!]_3$ using in total $2+6$ qutrits

5 Conclusions

We have presented two general methods for computing efficient encoding circuits for a stabilizer quantum code $\mathcal{C} = [\![n, k, d]\!]_q$. The first method uses only n qudits, while the second uses k input qudits and n output qudits simultaneously. The circuits obtained by the second method are closely related to graphs. Both methods yield circuits with overall complexity $O(n^2)$. In [9] we have presented a third method which yields a quantum circuit operating on n qudits in total.

The structure of that circuit is similar to the structure of the encoding circuit obtained by the first method presented in this article.

As neither the standard form of the stabilizer matrix used in the first method nor the graph related to the second method are unique, there is plenty of room for optimizations (see, e.g. [12]). We will address this question in future work.

Acknowledgments

The author would like to thank Martin Rötteler and Pradeep Sarvepalli for many fruitful discussions related to the present article. The Centre for Quantum Technologies is a Research Centre of Excellence funded by the National Research Foundation and the Ministry of Education.

References

1. Ashikhmin, A., Knill, E.: Nonbinary quantum stabilizer codes. IEEE Transactions on Information Theory 47(7), 3065–3072 (2001) (preprint quant-ph/0005008)
2. Calderbank, A.R., Rains, E.M., Shor, P.W., Sloane, N.J.A.: Quantum Error Correction Via Codes over GF(4). IEEE Transactions on Information Theory 44(4), 1369–1387 (1998) (preprint quant-ph/9608006)
3. Cleve, R., Gottesman, D.: Efficient computations of encodings for quantum error correction. Physical Review A 56(1), 76–82 (1997)
4. Gottesman, D.: A Class of Quantum Error-Correcting Codes Saturating the Quantum Hamming Bound. Physical Review A 54(3), 1862–1868 (1996)
5. Gottesman, D.: Stabilizer Codes and Quantum Error Correction. PhD thesis, California Institute of Technology, Pasadena, California (1997)
6. Gottesman, D.: Errata in "Stabilizer Codes and Quantum Error Correction" (2004), `http://www.perimeterinstitute.ca/personal/dgottesman/thesis-errata.html`
7. Grassl, M.: Algorithmic aspects of quantum error-correcting codes. In: Brylinski, R.K., Chen, G. (eds.) Mathematics of Quantum Computation, pp. 223–252. CRC Press, Boca Raton (2002)
8. Grassl, M., Klappenecker, A., Rötteler, M.: Graphs, Quadratic Forms, and Quantum Codes. In: Proceedings of the 2002 IEEE International Symposium on Information Theory, June 30 - July 5, p. 45. IEEE, Los Alamitos (2002) (preprint quant-ph/0703112)
9. Grassl, M., Rötteler, M., Beth, T.: Efficient Quantum Circuits for Non-Qubit Quantum Error-Correcting Codes. International Journal of Foundations of Computer Science (IJFCS) 14(5), 757–775 (2003) (preprint quant-ph/0211014)
10. Grassl, M., Rötteler, M., Beth, T.: On Quantum MDS codes. In: Proceedings of the 2004 IEEE International Symposium on Information Theory, Chicago, June 25 - July 2, p. 355. IEEE, Los Alamitos (2004)
11. Knill, E., Laflamme, R.: Theory of quantum error-correcting codes. Physical Review A 55(2), 900–911 (1997) (preprint quant-ph/9604034)

12. Kuo, K.Y., Lu, C.C.: A Further Study on the Encoding Complexity of Quantum Stabilizer Codes. In: Proceedings 2010 International Symposium on Information Theory and its Applications (ISITA), October 17-20, pp. 1041–1044 (2010)
13. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information. Cambridge University Press, Cambridge (2000)
14. Schlingemann, D.: Stabilizer codes can be realized as graph codes. Quantum Information & Computation 2(4), 307323 (2002) (preprint quant-ph/0111080)
15. Schlingemann, D., Werner, R.F.: Quantum error-correcting codes associated with graphs. Physical Review A 65(1), 012308 (2002) (preprint quant-ph/0012111)

Algorithms for the Shortest and Closest Lattice Vector Problems

Guillaume Hanrot, Xavier Pujol, and Damien Stehlé

Laboratoire LIP (U. Lyon, CNRS, ENS Lyon, INRIA, UCBL),
46 Allée d'Italie, 69364 Lyon Cedex 07, France
{guillaume.hanrot,xavier.pujol,damien.stehle}@ens-lyon.fr

Abstract. We present the state of the art solvers of the Shortest and Closest Lattice Vector Problems in the Euclidean norm. We recall the three main families of algorithms for these problems, namely the algorithm by Micciancio and Voulgaris based on the Voronoi cell [STOC'10], the Monte-Carlo algorithms derived from the Ajtai, Kumar and Sivakumar algorithm [STOC'01] and the enumeration algorithms originally elaborated by Kannan [STOC'83] and Fincke and Pohst [EUROCAL'83]. We concentrate on the theoretical worst-case complexity bounds, but also consider some practical facets of these algorithms.

1 Introduction

The Shortest Lattice Vector Problem (SVP) consists in finding $\boldsymbol{x} \in \mathbb{Z}^n \setminus \mathbf{0}$ minimizing $\|B \cdot \boldsymbol{x}\|$, where $B \in \mathbb{Q}^{m \times n}$ is given as input. The Closest Lattice Vector Problem (CVP) consists in finding $\boldsymbol{x} \in \mathbb{Z}^n$ minimizing $\|B \cdot \boldsymbol{x} - \boldsymbol{t}\|$, where $B \in \mathbb{Q}^{m \times n}$ and $\boldsymbol{t} \in \mathbb{Q}^m$ are given as inputs.[1] In this survey, we will restrict ourselves to the Euclidean norm $\|\boldsymbol{y}\| = \sqrt{\sum_{i \leq m} y_i^2}$. These optimization problems admit simple geometric interpretations: SVP consists in finding a shortest non-zero vector in the Euclidean lattice $L[B] := \sum_{i \leq n} x_i \boldsymbol{b}_i$ spanned by the columns $(\boldsymbol{b}_i)_i$ of B, whereas CVP consists in finding a vector of $L[B]$ closest to the given target $\boldsymbol{t}$.

SVP and CVP have been investigated in mathematics for more than a century, with, among others, the works of Hermite [38], Korkine and Zolotarev [46], Minkowski [59] and Voronoi [81]. However, the algorithmic study of lattices took off rather lately, at the beginning of the 1980's. At that time, lattices happened to be a bottleneck in combinatorial optimization and in algorithmic number theory, and ground-breaking results were then obtained: A. Lenstra, H. Lenstra Jr. and L. Lovász proposed the first polynomial-time approximation algorithm for SVP [47], whereas P. van Emde Boas showed that the decisional variant of CVP, is NP-hard [19] (as well as the decisional variant of SVP for the infinity norm). Shortly afterwards, Fincke and Pohst [20,21] and Kannan [42,43]

[1] Wlog we will assume that $\boldsymbol{t}$ belongs to the span of the columns of B, as otherwise it suffices to solve CVP for the orthogonal projection of $\boldsymbol{t}$ onto it.

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 159–190, 2011.
© Springer-Verlag Berlin Heidelberg 2011

described the first SVP and CVP solvers. Following the genesis of lattice-based cryptography in the mid-1990's [4,39,27], whose security provably/heuristically relies on the hardness of variants of SVP and CVP, much effort was spent devising faster solvers. This resulted in the construction of a new type of SVP and CVP algorithms by Ajtai, Kumar and Sivakumar [5,6]. More recently, yet another completely different algorithm was introduced by Micciancio and Voulgaris [57,56].

SVP and CVP are common in many fields of computational mathematics and computer science. We have already mentioned combinatorial optimization and algorithmic number theory. We refer the interested reader to the surveys [17] and [48]. They are also very frequent in communications theory [60,1,34,61]. The cryptographic role of SVP and CVP is twofold. In the last few years, a number of cryptographic primitives have been devised along with proofs of security under the assumption that there is no (probabilistic and sometimes quantum) polynomial-time algorithm for solving arbitrary instances of variants of SVP and CVP. Attempting to solve SVP and CVP allows to assess the validity of these assumptions. We refer to [55,72] for recent accounts on lattice-based cryptography. On the other hand, the best known algorithm for breaking these cryptographic schemes as well as a number of other cryptographic functions such as some based on the knapsack problem [66] attempt to find short or close vectors from a relevant lattice, by reducing it. Lattice reduction consists in starting from a basis B and trying to improve its quality, traditionally measured by the orthogonality of its vectors. The most famous lattice reduction algorithm is probably LLL (see the book [64]). It runs in polynomial time but it only provides vectors that are no more than exponentially longer (in n) than the shortest ones. This worst-case behavior seems to also hold in practice [63] (up to a constant factor in the exponent). LLL may also be used to find lattice vectors relatively close to target vectors [7], but these vectors may be exponentially further from the target than the optimal solution(s). Schnorr's hierarchy [74] of reduction algorithms allows to achieve a continuum between LLL and exact SVP and CVP solvers. The best known theoretical variant (in terms of achieved basis quality for any fixed computational cost) is due to Gama and Nguyen [23]. However, in practice, the heuristic and somewhat mysterious BKZ algorithm from [75] is used instead (see [24] for a detailed account on the practical behavior of BKZ). All known realizations of Schnorr's hierarchy (see the surveys [62,73]) rely on an algorithm that solves SVP for smaller-dimensional lattices.

From a computational hardness perspective, the decisional variant of CVP is known to be NP hard [19], whereas the decisional variant of SVP is only known to be NP hard under randomized reductions [2] (see also the book [54]). This remains the case for the relaxed variants GapSVP_γ and GapCVP_γ for any constant $\gamma \geq 1$, where the optimal value of $\|B \cdot \boldsymbol{x}\|$ (resp. $\|B \cdot \boldsymbol{x} - \boldsymbol{t}\|$) is known to be either ≤ 1 or $\geq \gamma$ (see [44,35]). When $\gamma = \Omega(\sqrt{n})$, GapSVP_γ and GapCVP_γ belong to NP $\cap$ coNP, and are thus unlikely to be NP hard [70]. Schnorr's hierarchy coupled with the SVP solver from [57,56] allows one to solve GapSVP_γ and GapCVP_γ for $\gamma = k^{O(n/k)}$ in time $2^{O(k)}$. In particular, the

relaxation factor $\gamma = 2^{c\frac{n \log\log n}{\log n}}$ can be achieved in polynomial time [74] for any constant $c > 0$. It is also worth noting that there is a dimension-preserving deterministic polynomial-time reduction from GapSVP_γ to GapCVP_γ for any γ (see [28]).

Three families of SVP and CVP solvers. There exist three main families of SVP and CVP solvers, which we compare in Table 1. Describing them is the purpose of the present survey.

The algorithm by Micciancio and Voulgaris [57,56] aims at computing the Voronoi cell of the lattice, whose knowledge facilitates the tasks of solving SVP and CVP. This algorithm allows one to solve SVP and CVP deterministically, in time $\leq 2^{2n+o(n)}$ and space $\leq 2^{n+o(n)}$. We will describe this algorithm in Section 3.

Table 1. Comparing the three families of SVP and CVP solvers

	Time complexity upper bound	Space complexity upper bound	Remarks
Sec. 3	$2^{2n+o(n)}$	$2^{n+o(n)}$	Deterministic
Sec. 4, SVP	$2^{2.465n+o(n)}$	$2^{1.325n+o(n)}$	Monte-Carlo
Sec. 4, CVP	$(2+1/\varepsilon)^{O(n)}$	$(2+1/\varepsilon)^{O(n)}$	Monte-Carlo solves $(1+\varepsilon)$-CVP only
Sec. 5, SVP	$n^{n/(2e)+o(n)}$	$\mathcal{P}oly(n)$	Deterministic
Sec. 5, CVP	$n^{n/2+o(n)}$	$\mathcal{P}oly(n)$	Deterministic

Singly exponential time complexity had already been achieved about 10 years before by Ajtai, Kumar and Sivakumar [5,6], with an algorithm that consists in saturating the space with a cloud of (perturbed) lattice points. But the saturation algorithms have at least three drawbacks: they are Monte Carlo (their success probability can be made exponentially close to 1, though), the CVP variants of these algorithms may only find vectors that are no more than $1+\varepsilon$ times further away from the target than the optimal solution, for an arbitrary $\varepsilon > 0$ (the complexity grows when ε tends to 0), and their best known complexity upper bounds are higher than that of the Voronoi-based Micciancio-Voulgaris algorithm. The saturation-based Ajtai *et al.* SVP solver has been successively improved in [69,65,58,68], and the currently best time complexity upper bound is $2^{2.465n+o(n)}$, with a space requirement bounded by $2^{1.325n+o(n)}$. Improvements on the Ajtai *et al.* CVP solver have been proposed by Blömer and Naewe [9]. We will describe the saturation-based SVP and CVP solvers in Section 4.

Before the algorithms of Ajtai *et al.*, the fastest SVP and CVP solvers relied on a deterministic procedure that enumerates all lattice vectors below a prescribed norm, or within a prescribed distance to a given target vector. This procedure uses the Gram-Schmidt orthogonalization of the input basis to recursively bound the integer coordinates of the candidate solutions. Enumeration-based SVP and CVP solvers were first described by Fincke and Pohst [20,21] and Kannan [42,43].

Kannan used it to propose solvers with bit-complexities $n^{O(n)}$. These were later refined by Helfrich [36], and their analyzes were improved in [32] who proved that the SVP (resp. CVP) solver has complexity $\leq n^{n/(2e)+o(n)}$ (resp. $\leq n^{n/2+o(n)}$). By refining a construction due to Ajtai [3], Hanrot and Stehlé [33] showed the existence of input bases for which Kannan's SVP algorithm performs $\geq n^{n/(2e)+o(n)}$ bit operations. We will describe these algorithms in Section 5.

Solving SVP and CVP in practice. The practicality of SVP solvers has attracted much more attention than their CVP counterparts, due to their importance in Schnorr's hierarchy [74] and their cryptanalytic applications. For currently handleable dimensions, the enumeration-based SVP solvers seem to outperform those of the other families. This statement requires clarification, as rigorous codes providing correctness guarantees can be accelerated significantly by allowing heuristics, which makes the comparison task more complex.

All the available implementations providing strong correctness guarantees (e.g., `fplll` [12] or the SVP solvers of Magma [10]) rely on the enumeration process. Several heuristics are known for the solvers of the saturation and enumeration families (at the moment, the Micciancio-Voulgaris Voronoi-based algorithm seems uncompetitive, and would require further practical investigation). However, the heuristic implementations of the enumeration families, relying on tree pruning strategies [75,76,80,25] seem to outperform the heuristic implementations of the saturation families [65,58]. This observation has led to hardware implementations of the enumeration [37,16].

It is hard to compare the soundness of the different heuristics used, but one way to compare the resulting codes is to see how they perform on actual inputs. However, checking the result is non-trivial, as there is no known way of rigorously doing so (apart from solving SVP once more). To circumvent this problem, the authors of the online SVP challenges [26] sampled SVP instances from some specific distribution [29] for which the expectancy of the minimum is known (via the Gaussian heuristic, see Section 5). If the vector found is no longer than slightly more than this expectancy, the challenge is considered solved. Another possibility, suggested in [24], consists in generating instances where the shortest vector is most likely the unique non-zero lattice vector whose norm is some known threshold (such as lattices corresponding to knapsacks [66]). Yet another strategy consists in looking at the reduction qualities achieved by using the SVP solvers within an implementation of Schnorr's hierarchy (typically BKZ). Comparisons are thus possible via lattice reduction challenges [49]. A drawback of this approach is that although the SVP solver is asymptotically the cost-dominating ingredient in reduction algorithms, there are other important factors, such as the number of times the SVP solver is called.

Related topics. Many problems on lattices are closely related to SVP and CVP. Sometimes but not always, the best algorithms for solving them are essentially the same as for solving SVP and CVP.

A variant of SVP consists in listing all shortest non-zero vectors, or counting them (their number is the so-called kissing number). The Theta series is a formal

series whose coefficient of degree k is the number of vectors of squared norm equal to k (for all $k \geq 0$). The first coefficients of the Theta series of a lattice are typically computed with enumeration-based algorithms (see, e.g., [80]).

Given a lattice L, it may be desirable to compute the successive minima $\lambda_i(L) = \min\{r : \dim(\text{span}(L \cap \mathcal{B}(r))) \geq i\}$, for all $i \leq n$, and linearly independent lattice vectors that reach them. Micciancio [53] showed how this problem can be reduced to CVP. Blömer and Naewe [9] have proposed a saturation-based algorithm that returns linearly independent vectors whose norms are not much greater than the minima.

The covering radius of a lattice is the maximal distance to that lattice of a vector of the spanned linear space. An algorithm relying on the Ajtai *et al.* CVP solver [31] provides a tight approximation to it. Also connected to CVP is the Bounded Distance Decoding problem (BDD): in BDD, the target vector is guaranteed to be within a distance $r \leq \lambda_1(L)/2$ of the lattice L. Several algorithms [45,50] have been specifically designed for such CVP instances. BDD is of particular significance for MIMO communications [61] and cryptography [71]. We also refer to [51] for reductions of several lattice problems to and from BDD.

In this survey, we only consider the Euclidean norm. Recent works on solving SVP/CVP for the other ℓ_p norms include [9,15,18].

2 Some Background on Euclidean Lattices

A lattice is a discrete additive subgroup of some $\mathbb{R}^m$. Any such object can be represented as $\{B \cdot \boldsymbol{x}, \boldsymbol{x} \in \mathbb{Z}^n\}$, for some n and some full column rank matrix $B \in \mathbb{R}^{m \times n}$. For a given lattice L, some quantities, or invariants, do not depend on the particular choice of the basis. These include, among others: the embedding dimension m, the lattice rank n, the lattice determinant $\det L = \sqrt{\det(B^t \cdot B)}$, the lattice minimum $\lambda(L) = \min_{\boldsymbol{x} \in \mathbb{Z}^n \setminus \mathbf{0}} \|B \cdot \boldsymbol{x}\|$ and the covering radius $\mu(L) = \max_{\boldsymbol{t} \in \text{span}(L)}(\text{dist}(\boldsymbol{t}, L))$. We refer to [13,30,78,52] for mathematical introductions to lattices. The most fundamental result on lattice invariants is Minkowski's theorem, which states that for all n, there exists a constant $\gamma_n \leq n$ such that for any rank n lattice L, we have $\lambda(L) \leq \sqrt{\gamma_n}(\det L)^{1/n}$.

When $L = \{B \cdot \boldsymbol{x}, \boldsymbol{x} \in \mathbb{R}^n\}$ with a full column rank matrix B, we say that the columns $(\boldsymbol{b}_i)_i$ of B form a basis of the lattice L. Any given lattice of rank ≥ 2 has infinitely many lattice bases: The columns of the full column rank matrices $B_1, B_2 \in \mathbb{R}^{m \times n}$ span the same lattice if and only if there exists a matrix $U \in \mathbb{Z}^{n \times n}$ of determinant ± 1 such that $B_2 = B_1 \cdot U$. Such a matrix U is called unimodular.

For computational statements, we will always consider rational lattices, given by rational bases (see [11] for the case of real-valued input bases). In the case of CVP, the target vector will also be rational. The complexities of the algorithms we will study are typically exponential with respect to n but only polynomial with respect to m and the bit-sizes of the entries of the input matrix. For the sake of simplicity, we will assume that the bit-size of the input is $\mathcal{P}oly(n)$.

The process of improving the quality of a basis by applying well-chosen unimodular matrices is generically called lattice reduction. The achieved qualities, or reductions, are most often defined in terms of orthogonality. For a given basis $(\boldsymbol{b}_i)_{i\leq n}$, we define its Gram-Schmidt orthogonalization by $\boldsymbol{b}_i^* = \boldsymbol{b}_i - \sum_{j<i} \mu_{i,j}\boldsymbol{b}_j^*$, where $\mu_{i,j} = \langle \boldsymbol{b}_i, \boldsymbol{b}_j^* \rangle / \|\boldsymbol{b}_j^*\|^2$. As the $\boldsymbol{b}_i^*$'s are orthogonal, we have $\lambda(L) \geq \min_{i\leq n} \|\boldsymbol{b}_i^*\|$, where L is the lattice spanned by the $\boldsymbol{b}_i$'s. We also have $\mu(L) \leq \sqrt{\sum_{i\leq n} \|\boldsymbol{b}_i^*\|^2}/2$ (see [7]).

A basis is said size-reduced if $|\mu_{i,j}| \leq 1/2$ for any $i > j$. A basis is said LLL-reduced if it is size-reduced and if $\|\boldsymbol{b}_i^*\| \geq \|\boldsymbol{b}_{i-1}^*\|/2$ for any $i < n$. A LLL-reduced basis of any rational lattice can be computed in polynomial time from any input basis [47]. However, its quality is only moderate: the quantity $\|\boldsymbol{b}_1\|/\lambda(L)$ may be bounded by 2^n. This upper bound seems reached in practice, up to a constant factor in the exponent [63].

On the other hand, the Hermite-Korkine-Zolotarev (HKZ) reduction is more expensive to compute (it is polynomial-time equivalent to solving SVP), but HKZ-reduced bases are much more orthogonal. A basis $B = (\boldsymbol{b}_i)_{i\leq n}$ is said HKZ-reduced if it is size-reduced, if $\|\boldsymbol{b}_1\| = \lambda(L(B))$ and if once projected orthogonally to $\boldsymbol{b}_1$ the vectors $\boldsymbol{b}_2, \ldots, \boldsymbol{b}_n$ are themselves HKZ-reduced. By applying Minkowski's theorem on the projected lattices, it is possible to show that any HKZ-reduced basis $(\boldsymbol{b}_i)_i$ satisfies $\|\boldsymbol{b}_1\|/\lambda(L) \leq \exp((\log^2 n)/4)$ (see [33]).

Schnorr's hierarchy of reductions and reduction algorithms [74] offers a compromise between LLL's efficiency and HKZ's quality. The principle is to consider small dimensional projected blocks: if $(\boldsymbol{b}_i)_{i\leq n}$ is the current lattice basis, the β-dimensional projected block starting at index k is $(\boldsymbol{b}_i^{(k)})_{k\leq i<k+\beta}$, where $\boldsymbol{b}_i^{(k)} = \boldsymbol{b}_i - \sum_{j<k} \mu_{i,k}\boldsymbol{b}_j^*$ is the projection of $\boldsymbol{b}_i$ orthogonally to the span of the first $k-1$ basis vectors. Within blocks, one performs HKZ-reductions (or calls to an SVP solver), and blocks are handled in a LLL-manner. This vague description has been instantiated in several precisely analyzed hierarchies of reduction algorithms [74,22,23] and is also the basis of the famous heuristic BKZ algorithm [75] implemented in NTL [77]. The best complexity/quality trade-off currently known [23] allows one to find a basis $(\boldsymbol{b}_i)_i$ such that $\|\boldsymbol{b}_1\|/\lambda(L) \leq (2\gamma_\beta)^{\frac{n-\beta}{\beta-1}}$ using calls to an SVP solver in dimension β. Note that the practical quality of BKZ-reduced bases has been investigated in [24], and that worst-case quality lower bounds for fixed block-sizes are known [3,33]. Finally, in order to maximize $\|\boldsymbol{b}_n^*\|$ rather than minimize $\|\boldsymbol{b}_1\|$, one may reduce the dual basis B^{-t} (we refer to [69] for an introduction on the dual lattice), and apply the obtained unimodular transform to B. Thanks to Banaszczyk's transference theorem [8], the corresponding basis $(\boldsymbol{c}_i)_{i\leq n}$ of L satisfies $\mu(L)/\|\boldsymbol{c}_n^*\| \leq \gamma_n(2\gamma_\beta)^{\frac{n-\beta}{\beta-1}}$.

3 An SVP/CVP Solver Relying on the Voronoi Cell

The Micciancio-Voulgaris deterministic algorithm based on the Voronoi cell [57,56] provides the SVP/CVP solver with the best known complexity upper bound: It terminates within $2^{2n+o(n)}$ operations while its space requirement is $\leq 2^{n+o(n)}$.

From a practical perspective, this algorithm seems bound to remain slower than the guaranteed implementations of the enumeration-based solvers: On the one hand, its cost cannot be lower than 2^n, as this is the size of the representation of a generic Voronoi cell; On the other hand, although their time complexity bounds are asymptotically much higher than 2^n, the enumeration-based solvers still terminate relatively efficiently in dimensions n higher than 60. The asymptotic bounds suggest Voronoi-based solvers will eventually beat enumeration-based solvers, but the cut-off dimension seems to be out of reach with nowadays computational power. Also, for the moment, there is no known heuristic version of the Micciancio-Voulgaris algorithm which would allow to break the 2^n barrier.

The Voronoi cell $\mathcal{V}(L)$ of a lattice L is the set of vectors strictly closer to the origin than to any other lattice point:

$$\mathcal{V}(L) = \{\boldsymbol{x} : \forall \boldsymbol{b} \in L, \|\boldsymbol{b} - \boldsymbol{x}\| > \|\boldsymbol{x}\|\}. \tag{1}$$

We let $\overline{\mathcal{V}}(L)$ denote the topological closure of $\mathcal{V}(L)$, i.e., the set of points closer or at equal distance to the origin than to any other lattice point.

The definition (1) involves an infinite number of inequalities. In fact, there exists a minimal set $(\boldsymbol{v}_j)_{j \leq m}$ of vectors of L that suffices to define $\mathcal{V}(L)$: $\mathcal{V}(L) = \{\boldsymbol{x} : \forall j \leq m, \|\boldsymbol{v}_j - \boldsymbol{x}\| > \|\boldsymbol{x}\|\}$. Stated differently, the Voronoi cell is the interior of a polytope. We call these vectors the relevant vectors of L. Voronoi [81] displayed a strong link between the relevant vectors and the cosets of $L/2L$. A coset of $L/2L$ is of the shape $\{\sum_i (2x_i + e_i)\boldsymbol{b}_i, x_i \in \mathbb{Z}\}$, where $(\boldsymbol{b}_i)_i$ is a basis of L and the e_i's are fixed elements of $\{0, 1\}$. The vector $\boldsymbol{e}$ may be interpreted as the parity of the coset (note that it depends on the choice of the lattice basis $(\boldsymbol{b}_i)_i$).

Lemma 3.1 ([14, Th. 10, p. 477]). *A vector $\boldsymbol{v} \in L \setminus 2L$ is a relevant vector of the lattice L if $\pm\boldsymbol{v}$ are the unique minima (for the norm) of the coset $\boldsymbol{v} + 2L$. Consequently, there are $\leq 2(2^n - 1)$ relevant vectors.*

3.1 The Voronoi Cell Suffices for Solving SVP and CVP

Assume we know the relevant vectors $(\boldsymbol{v}_i)_{i \leq m}$ of a lattice L. We now explain how to solve SVP and CVP by using this data.

To solve SVP in time $\mathcal{P}oly(n) \cdot 2^n$ from the $\boldsymbol{v}_i$'s, it suffices to observe that the shortest relevant vectors reach the lattice minimum.

Lemma 3.2. *If $\boldsymbol{s} \in L$ satisfies $\|\boldsymbol{s}\| = \lambda(L)$, then $\boldsymbol{s}$ is a relevant vector.*

Proof. The vector $\boldsymbol{s}$ cannot belong to $2L$ as otherwise $\boldsymbol{s}/2$ would be a shorter non-zero lattice vector. It remains to show that $\pm\boldsymbol{s}$ are the unique minima of the coset $\boldsymbol{s} + 2L$. Assume that $\|\boldsymbol{s} + 2\boldsymbol{b}\| = \|\boldsymbol{s}\|$ for some $\boldsymbol{b} \in L \setminus \boldsymbol{0}$. We have $\|\boldsymbol{s}\|^2 + 4\|\boldsymbol{b}\|^2 + 4\langle \boldsymbol{s}, \boldsymbol{b} \rangle = \|\boldsymbol{s}\|^2$, which leads to $\|\boldsymbol{b}\|^2 = -\langle \boldsymbol{s}, \boldsymbol{b} \rangle$. The Cauchy-Schwarz inequality provides $\|\boldsymbol{b}\| = \|\boldsymbol{s}\|$ if $\boldsymbol{b}$ and $\boldsymbol{s}$ are linearly dependent, and $\|\boldsymbol{b}\| < \|\boldsymbol{s}\|$ otherwise. In the first situation, the only way of ensuring that $\|\boldsymbol{s} + 2\boldsymbol{b}\| = \|\boldsymbol{s}\|$ is to take $\boldsymbol{b} = -\boldsymbol{s}$. In the other situation, we must have $\boldsymbol{b} = \boldsymbol{0}$, since $\boldsymbol{s}$ is a shortest non-zero vector of L. □

```
Input  : The relevant vectors (v_i)_{i≤N} and t ∈ 2V̄(L).
Output : A vector t' ∈ (t + L) ∩ V̄(L).
While t ∉ V̄(L) do
    Find i ≤ N maximizing ⟨t, v_i⟩/‖v_i‖², t ← t − v_i.
Return v.
```

Algorithm 1: The $\mathrm{CVP}_{2\overline{\mathcal{V}}\to\overline{\mathcal{V}}}$ algorithm.

We now explain how to solve CVP, i.e., subtract from a given target vector a lattice vector so that the result belongs to the closed Voronoi cell. The engine of the algorithm is a sub-routine $\mathrm{CVP}_{2\overline{\mathcal{V}}\to\overline{\mathcal{V}}}$ that solves CVP assuming that the target vector $\boldsymbol{t}$ belongs to $2\cdot\overline{\mathcal{V}}(L)$. If such a routine is available, then CVP can be solved using a polynomial number of calls to it. First, note that any target vector $\boldsymbol{t}$ in the linear span of L belongs to $2^k\cdot\overline{\mathcal{V}}(L)$, with $k=\log_2(2\|\boldsymbol{t}\|/\lambda(L))$ (because $\mathcal{B}(\lambda(L)/2)\subseteq\overline{\mathcal{V}}(L)$). Now, the routine $\mathrm{CVP}_{2\overline{\mathcal{V}}\to\overline{\mathcal{V}}}$ may be used to subtract vectors of L to any given target vector $\boldsymbol{t}\in 2^\ell\cdot\overline{\mathcal{V}}(L)$ so that the result belongs to $(\boldsymbol{t}+L)\cap 2^{\ell-1}\cdot\overline{\mathcal{V}}(L)$: It suffices to use $\mathrm{CVP}_{2\overline{\mathcal{V}}\to\overline{\mathcal{V}}}$ with the lattice $2^{\ell-1}L$, whose closed Voronoi cell is $2^{\ell-1}\cdot\overline{\mathcal{V}}(L)$.

The sub-routine $\mathrm{CVP}_{2\overline{\mathcal{V}}\to\overline{\mathcal{V}}}$, is an improvement of the Iterative Slicer from [79]. Given a target $\boldsymbol{t}\in 2\cdot\overline{\mathcal{V}}$, it first determines if $\boldsymbol{t}$ belongs to $\overline{\mathcal{V}}(L)$, in which case it stops. Otherwise, it finds $x\in(1,2]$ such that $\boldsymbol{t}$ is on the boundary of $x\cdot\overline{\mathcal{V}}$, as well as a facet of $x\cdot\overline{\mathcal{V}}$ containing $\boldsymbol{t}$. Geometrically, it means that for every facet of $\overline{\mathcal{V}}$ we construct a cone of apex $\mathbf{0}$ and base that facet. These cones cover the whole space (and their interiors are disjoint), and we determine a cone containing $\boldsymbol{t}$. In practice, this is obtained by finding a relevant vector $\boldsymbol{v}_i$ that maximizes $\langle\boldsymbol{t},\boldsymbol{v}_i\rangle/\|\boldsymbol{v}_i\|^2$ (in which case $x=2\langle\boldsymbol{t},\boldsymbol{v}_i\rangle/\|\boldsymbol{v}_i\|^2$). Once $\boldsymbol{v}_i$ is found, the target $\boldsymbol{t}$ is updated to a new target vector $\boldsymbol{t}\leftarrow\boldsymbol{t}-\boldsymbol{v}_i$. This process is repeated until $\boldsymbol{t}$ eventually happens to belong to $\overline{\mathcal{V}}(L)$. Figure 1 provides an example of an execution of $\mathrm{CVP}_{2\overline{\mathcal{V}}\to\overline{\mathcal{V}}}$ in dimension 2. The following lemma states some important properties satisfied by the sequence of target vectors $(\boldsymbol{t}_k)_{k\geq 1}$ with $\boldsymbol{t}_1=\boldsymbol{t}$, corresponding to the successive loop iterations.

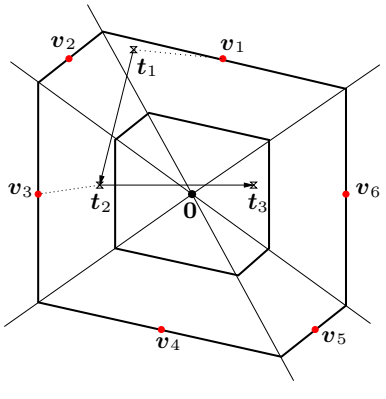


Fig. 1. Applying $\mathrm{CVP}_{2\overline{\mathcal{V}}\to\overline{\mathcal{V}}}$ to $\boldsymbol{t}_1$ with relevant vectors $(\boldsymbol{v}_i)_i$

Lemma 3.3. *Let $\boldsymbol{t}_1 \in 2 \cdot \overline{\mathcal{V}}(L)$ and $(\boldsymbol{t}_k)_{k \geq 1}$ be the sequence obtained by applying the algorithm $\mathrm{CVP}_{2\overline{\mathcal{V}} \to \overline{\mathcal{V}}}$ to $\boldsymbol{t}_1$. Then:*

- *For any k, we have $\boldsymbol{t}_k \in 2 \cdot \overline{\mathcal{V}}(L)$ and $\|\boldsymbol{t}_k\| > \|\boldsymbol{t}_{k+1}\|$,*
- *The execution terminates within $N \leq 2^n$ iterations and $\boldsymbol{t}_N \in \overline{\mathcal{V}}(L)$.*

Furthermore, if $\boldsymbol{t}_1 \in 2 \cdot \mathcal{V}(L)$, then $\boldsymbol{t}_k \in 2 \cdot \mathcal{V}(L)$ for all k.

Proof. The first claim derives from the observation that both $\boldsymbol{t}_k$ and $\boldsymbol{t}_k - x\boldsymbol{v}_{i(k)}$ belong to $x \cdot \overline{\mathcal{V}}(L) \subseteq 2 \cdot \overline{\mathcal{V}}(L)$ (because $\frac{1}{x}\boldsymbol{t}_k$ belongs to both $\overline{\mathcal{V}}(L)$ and $\boldsymbol{v}_{i(k)} + \overline{\mathcal{V}}(L)$). By convexity, so does $\boldsymbol{t}_k - \boldsymbol{v}_{i(k)}$. For the second claim, note that by construction we have $\|\boldsymbol{t}_k - x\boldsymbol{v}_{i(k)}\| = \|\boldsymbol{t}_k\|$. This implies that $\|\boldsymbol{t}_k - \boldsymbol{v}_{i(k)}\|^2 = \|\boldsymbol{t}_k\|^2 - (x - 1)\|\boldsymbol{v}_{i(k)}\|^2 < \|\boldsymbol{t}_k\|^2$. At this stage, we have proved that we have a sequence of vectors of $(\boldsymbol{t}_1 + L) \cap 2 \cdot \overline{\mathcal{V}}(L)$ of strictly decreasing norms. Each such vector belongs to one of the 2^n cosets of $\boldsymbol{t}_1 + L$ modulo $2L$. We show by contradiction that such a coset cannot be visited twice. Let $k < \ell$ such that $\boldsymbol{t}_k = \boldsymbol{t}_\ell \bmod 2L$. Since both $\boldsymbol{t}_k$ and $\boldsymbol{t}_\ell$ belong to $\overline{\mathcal{V}}(2L)$, they must be shortest elements of their respective cosets $\boldsymbol{t}_k + 2L$ and $\boldsymbol{t}_\ell + 2L$. Since we assumed that these cosets are equal, the vectors $\boldsymbol{t}_k$ and $\boldsymbol{t}_\ell$ must have equal norms. This is in contradiction with $\|\boldsymbol{t}_k\| > \|\boldsymbol{t}_\ell\|$. □

This result allows us to bound the cost of sub-routine $\mathrm{CVP}_{2\overline{\mathcal{V}} \to \overline{\mathcal{V}}}$ by $\mathcal{P}oly(n) \cdot 2^{2n}$. Overall, once the relevant vectors of L are known, it is possible to solve CVP within $\mathcal{P}oly(n) \cdot 2^{2n}$ bit operations.

3.2 Computing the Relevant Vectors

In the SVP and CVP solvers of the previous subsection, the list of the relevant vectors of the lattice L under scope was assumed known. We now explain how to obtain such a list.

Let $(\boldsymbol{b}_i)_{i \leq n}$ be a basis of L that such that $\|\boldsymbol{b}_n^*\| \geq 2\mu(L)/\phi_n$, for some known $\phi_n = 2^{o(n)}$. This can be achieved by applying Schnorr's hierarchy with block-size $\omega(1)$ (see Section 2), on the dual basis of $(\boldsymbol{b}_i)_{i \leq n}$. The purpose of this pre-processing of the lattice basis will become clearer soon.

We assume we know the relevant vectors of $L^- := L[\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{n-1}]$ (thanks to a recursive call in dimension $n - 1$). A relevant vector of L is a strict length minimum for a coset of $L/2L$: it cannot be made smaller by subtracting from it an element of $2L$. In particular, it cannot be made smaller by subtracting from it a relevant vector of $2L^-$, i.e., twice a relevant vector of L^-. This implies that all the relevant vectors of L are contained in the interior of the cylinder of basis $2 \cdot \mathcal{V}(L^-)$ that is orthogonal to the span of L^-.

We now show that the relevant vectors of L cannot be arbitrarily far away from the span of L^-, thus allowing us to bound the search region. Let $\boldsymbol{t} = \sum_i e_i \boldsymbol{b}_i$ with $e_i \in \{0, 1\}$ for all i. We search for the relevant vector $\sum(e_i + 2x_i)\boldsymbol{b}_i \in L$ corresponding to the coset $\boldsymbol{t} + 2L$. This is a CVP instance for the target $\boldsymbol{t}$ and the lattice $2L$. We cannot use the CVP algorithm of the previous subsection

directly, as it requires the knowledge of the relevant vectors of L, which we are currently trying to compute. Instead, we note that:

$$\mathrm{CVP}(\boldsymbol{t}, 2L) = \min_{\|\cdot\|} \left\{ \mathrm{CVP}(\boldsymbol{t} + 2x_n\boldsymbol{b}_n, 2L^-) : x_n \in \mathbb{Z} \right\}.$$

In fact, since $\boldsymbol{t}$ is within distance $2\mu(L)$ from $2L$ and $\|\boldsymbol{t}+2x_n\boldsymbol{b}_n+2\sum_{i<n} x_i\boldsymbol{b}_i\| \geq |e_n + 2x_n|\|\boldsymbol{b}_n^*\|$, we obtain that it suffices to consider the x_n's such that $|e_n + 2x_n| \leq 2\mu(L)/\|\boldsymbol{b}_n^*\|$. Thanks to the reducedness of the $\boldsymbol{b}_i$'s, it suffices to consider a bounded number of x_n's. This dimension reduction trick for CVP is inspired from the enumeration-based CVP solver (see Section 5). Overall, we have proved the following result.

Lemma 3.4. *Let $(\boldsymbol{b}_i)_i$ be a basis of a lattice L with $\|\boldsymbol{b}_n^*\| \geq 2\mu(L)/\phi_n$. Then the relevant vectors of L are contained in the set*

$$\bigcup_{|x_n| \leq \phi_n} \left[x_n\boldsymbol{b}_n^* + \left(x_n(\boldsymbol{b}_n - \boldsymbol{b}_n^*) + L^- \cap 2 \cdot \mathcal{V}(L^-) \right) \right],$$

where $L^- = L[\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{n-1}]$.

This justifies Algorithm 2. Its cost is no more than $2\phi_n + 1 = 2^{o(n)}$ times the cost of enumerating $(\boldsymbol{t} + L^-) \cap 2 \cdot \mathcal{V}(L^-)$, for an arbitrary $\boldsymbol{t}$ in the span of the lattice L^-, whose relevant vectors are known.

`Input` : A basis of a lattice L.
`Output` : The relevant vectors of L.
Find a basis $(\boldsymbol{b}_i)_{i \leq n}$ of L such that $\|\boldsymbol{b}_n^*\| \geq 2\mu(L)/\phi_n$.
If $n = 1$, then Return $\pm\boldsymbol{b}_1$.
Compute the relevant vectors $(\boldsymbol{v}_i)_{i \leq N}$ of L^-.
`Cand` $\leftarrow \emptyset$. For $x_n = -\phi_n$ to ϕ_n do
    $(\boldsymbol{t}_i)_i \leftarrow \mathtt{Enum}_{2\mathcal{V}}(x_n(\boldsymbol{b}_n - \boldsymbol{b}_n^*), L^-)$,
    `Cand` $\leftarrow$ `Cand` $\cup \{x_n\boldsymbol{b}_n^* + \boldsymbol{t}_i\}_i$.
`Vor` $\leftarrow \emptyset$. For any non-zero coset $\mathcal{C}$ of $L/2L$ do
    If there are strict minima $\pm\boldsymbol{v}$ in $\mathcal{C} \cap$ `Cand`, then add them to `Vor`.
Return `Vor`.

Algorithm 2: Computing the relevant vectors

We now explain how to draw the list of all the elements belonging to $(\boldsymbol{t} + L^-) \cap 2 \cdot \mathcal{V}(L^-)$. Algorithm $\mathtt{Enum}_{2\mathcal{V}}$ first computes a shortest representative $\boldsymbol{s}$ of $\boldsymbol{t} + L^-$: If $\boldsymbol{t} = \boldsymbol{0}$, then we have $\boldsymbol{s} = \boldsymbol{0}$, and otherwise we may use the CVP algorithm from the last subsection (recall that we know the relevant vectors of L^-). We prove the correctness of $\mathtt{Enum}_{2\mathcal{V}}$ only for the situation where $\boldsymbol{s}$ is the only shortest element in $\boldsymbol{t} + L^-$. Correctness also holds without this assumption and can be proved using results from [40] on the relevant vectors of a Voronoi cell [82]. A proof will appear in the final version of [57,56].

The next step of $\mathtt{Enum}_{2\mathcal{V}}$ consists in backtracking all the possible executions of $\mathrm{CVP}_{2\overline{\mathcal{V}} \to \overline{\mathcal{V}}}$ that start from an element of $(\boldsymbol{t} + L^-) \cap 2 \cdot \mathcal{V}(L^-)$: If $\boldsymbol{s}$ is the

unique shortest element in $\boldsymbol{t}+L^-$, then any of these executions has to terminate with $\boldsymbol{s}$. Algorithm $\texttt{Enum}_{2\mathcal{V}}$ starts from a list of accessible vectors $\texttt{Acc}$ consisting of $\boldsymbol{s}$ and an empty list of visited vectors $\texttt{Vis}$. At any iteration, it takes the shortest accessible vector $\boldsymbol{u}$, tags it as visited, and adds to the accessible list the $\boldsymbol{u}+\boldsymbol{v}_i$'s for all relevant vectors $\boldsymbol{v}_i$. Before proceeding to the next iteration, $\texttt{Enum}_{2\mathcal{V}}$ cleans the accessible list by deleting all vectors belonging to a coset of $\boldsymbol{t}+L^-$ mod $2L^-$ reached by the visited list, and keeping a shortest accessible vector for each remaining coset of $\boldsymbol{t}+L^-$ mod $2L^-$. The process stops when the accessible list is empty. The number of iterations is bounded by the number of cosets of $\boldsymbol{t}+L^-$ mod $2L^-$, i.e., termination occurs within 2^{n-1} iterations, allowing us to bound the cost by $\mathcal{P}oly(n)\cdot 2^{2n}$.

$\texttt{Input}$: The relevant vectors $(\boldsymbol{v}_i)_{i\le N}$ of a lattice L^-, and $\boldsymbol{t}$.
$\texttt{Output}$: A superset of $(\boldsymbol{t}+L^-)\cap 2\cdot\mathcal{V}(L^-)$.
Find a shortest element $\boldsymbol{s}$ of $\boldsymbol{t}+L^-$.
$\texttt{Acc}\leftarrow\{\boldsymbol{s}\}$, $\texttt{Vis}\leftarrow\emptyset$. While $\texttt{Acc}\neq\emptyset$ do
  Let $\boldsymbol{u}$ be a shortest element of $\texttt{Acc}$,
  $\texttt{Vis}\leftarrow\texttt{Vis}\cup\{\boldsymbol{u}\}$, $\texttt{Acc}\leftarrow\texttt{Acc}\cup\{\boldsymbol{u}+\boldsymbol{v}_i\}_i$,
  For any coset $\mathcal{C}$ of $\boldsymbol{t}+L^-$ mod $2L^-$ do
    If $\mathcal{C}\cap\texttt{Vis}\neq\emptyset$, then $\texttt{Acc}\leftarrow\texttt{Acc}\setminus\mathcal{C}$,
    Else $\texttt{Acc}\leftarrow(\texttt{Acc}\setminus\mathcal{C})\cup\min_{\|\cdot\|}(\texttt{Acc}\cap\mathcal{C})$.
Return $\texttt{Vis}$.

Algorithm 3: The $\texttt{Enum}_{2\mathcal{V}}$ algorithm

Lemma 3.5. *Let L^- be a lattice and $\boldsymbol{t}\in\mathrm{span}(L^-)$. Suppose that there exists a unique shortest element $\boldsymbol{s}$ in $\boldsymbol{t}+L$. Then, at the end of the execution of $\texttt{Enum}_{2\mathcal{V}}$ for target $\boldsymbol{t}$ and lattice L^-, the set of visited vectors contains all elements of $\boldsymbol{t}+L^-\cap 2\cdot\mathcal{V}(L^-)$.*

Proof. Suppose by contradiction that $\boldsymbol{u}_1\in\boldsymbol{t}+L^-\cap 2\cdot\mathcal{V}(L^-)$ does not belong to the final visited list. By Lemma 3.3, there exists a sequence $\boldsymbol{u}_1,\boldsymbol{u}_2,\ldots,\boldsymbol{u}_N$ of vectors of $\boldsymbol{t}+L^-\cap 2\cdot\mathcal{V}(L^-)$ with $\boldsymbol{u_N}=\boldsymbol{s}$ and, for all k, $\|\boldsymbol{u}_k\|>\|\boldsymbol{u}_{k+1}\|$ and any $\boldsymbol{u}_{k+1}-\boldsymbol{u}_k$ is a relevant vector of L^-. Since $\boldsymbol{s}$ belongs to the final visited list, there must exist a $k<N$ such that $\boldsymbol{u}_{k+1}$ is in the visited list, but not $\boldsymbol{u}_k$. When $\boldsymbol{u}_{k+1}$ was included in the visited list, the vector $\boldsymbol{u}_k$ was added in the accessible list, because $\boldsymbol{u}_{k+1}-\boldsymbol{u}_k$ is a relevant vector. Since $\boldsymbol{u}_k$ is the sole vector of its coset of $\boldsymbol{t}+L^-$ mod $2L^-$ that belongs to $\mathcal{V}(2L^-)$, it cannot have been discarded later. This contradicts the definition of k. □

To conclude, the cost of computing the relevant vectors for the n-dimensional lattice L is that of sufficiently reducing the input basis of L, computing the relevant vectors of L^- of dimension $n-1$ and then running $\texttt{Enum}_{2\mathcal{V}}$ no more than $2\phi_n+1=2^{o(n)}$ times. This leads to the claimed $2^{2n+o(n)}$ complexity upper bound.

4 Saturating the Space

In this section, we describe the category of sieve algorithms for SVP and (approximate) CVP, which are Monte-Carlo probabilistic algorithms running in exponential time and space. Heuristic versions of these algorithms have been used to solve SVP up to dimension 63 [65,58].

The mathematical property which all the sieve algorithms try to exploit is that there are boundably many points within a compact body which are distant from one another. Thus, by saturating the space with (possibly perturbed) lattice points, one eventually finds close-by or identical vectors. In Euclidean norm, the proof of the saturation property leading the best known complexity bounds relies on the following result on sphere packings.

Theorem 4.1 ([41]). *Let $E \subseteq \mathbb{R}^n \backslash \{\mathbf{0}\}$. If there exists $\phi_0 > 0$ such that for any $\boldsymbol{u}, \boldsymbol{v} \in E$, the angle between $\boldsymbol{u}$ and $\boldsymbol{v}$ is $\geq \phi_0$, then $|E| \leq 2^{cn+o(n)}$ with $c = -\frac{1}{2}\log_2\left[1 - \cos(\min(\phi_0, 62.99^\circ))\right] - 0.099$.*

4.1 The AKS Algorithm

The `AKS` algorithm was introduced by Ajtai, Kumar and Sivakumar in [5] as the first single-exponential time algorithm for SVP. However, no explicit time bound was given. In [69], Regev described a simpler version of this algorithm running in time $2^{16n+o(n)}$. The constant in the exponent was decreased from 16 to 5.9 by Nguyen and Vidick [65], 3.4 by Micciancio and Voulgaris [58] and 2.7 by Pujol and Stehlé [68].

`AKS` can be described as follows: Let $\gamma < 1$ be a constant. Let $\mathcal{S}$ be a set of N lattice vectors sampled in the ball of radius $R = 2^{O(n)} \cdot \lambda(L)$. If N is large enough, there exists a pair $(\boldsymbol{u}, \boldsymbol{v})$ of vectors such that $\|\boldsymbol{u} - \boldsymbol{v}\| \leq \gamma R$, so $\boldsymbol{u} - \boldsymbol{v}$ is a shorter vector of L. The main step of the algorithm, called sieve, consists in choosing $\mathcal{C} \subseteq \mathcal{S}$ such that $|\mathcal{C}|$ is not too large and for any $\boldsymbol{u} \in \mathcal{S} \setminus \mathcal{C}$, there exists $\boldsymbol{v} \in \mathcal{C}$ such that $\|\boldsymbol{u} - \boldsymbol{v}\| \leq \gamma R$ (see Figure 2). This is used to generate a set $\mathcal{S}' \subseteq L \cap \mathcal{B}(\gamma R)$ with $|\mathcal{S}'| = |\mathcal{S}| - |\mathcal{C}|$. The sieve can be applied a polynomial number of times to obtain lattice vectors whose norms are $\leq r_0 \lambda(L)$ for some constant r_0.

There is no known way of ensuring that the vectors in the final set are uniformly distributed, but a technical trick allows to ensure that any shortest non-zero vector of L can be written as the difference between two vectors in the final set. A perturbation randomly chosen in $\mathcal{B}(\xi\lambda(L))$ is added to each sampled lattice vector to obtain a perturbed vector (if $\lambda(L)$ is unknown, one may try polynomially many guesses). At any time, the algorithm keeps track of both lattice vectors and perturbed vectors, applying the same operations on them. Provided that $\xi > \frac{1}{2}$, a given perturbed vector might sometimes correspond to two different lattice vectors whose distance is $\lambda(L)$. The sieve function makes

tests only on perturbed vectors so is unaware of the genuine lattice vectors. This can be used to prove that the probability of success is at most $2^{O(n)}$ times smaller than the probability of having the same lattice vector twice in the final set. The latter occurs when it contains $\geq |L \cap \mathcal{B}(r_0\lambda(L))|$ elements.

The algorithm is given below in the version of [65]. Perturbations are generated before lattice vectors and are used to compute them. In particular, if $\boldsymbol{x}$ and $\boldsymbol{y}$ are two perturbations such that $\boldsymbol{x} - \boldsymbol{y} \in L$, they will lead to the same lattice vector. Although this is might look as the most natural solution, this idea introduced by Regev [69] makes the proof simpler. In algorithm `NewPair`, the vector $(-\boldsymbol{x})$ mod $\mathcal{P}(B)$ is the vector whose coordinates with respect to B are the fractional part of the coordinates of $-\boldsymbol{x}$ with respect to B.

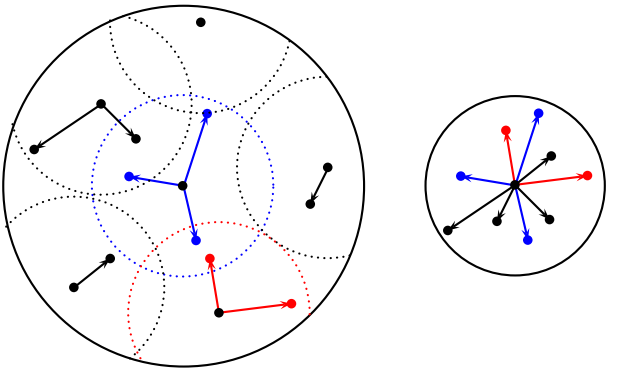


Fig. 2. The set $\mathcal{S}$ before and after one step of the sieve

`Input` : A basis B and a perturbation $\boldsymbol{x}$.
`Output` : A lattice vector $\boldsymbol{u}$ and a perturbed vector $\boldsymbol{u}'$.
$\boldsymbol{u}' \leftarrow (-\boldsymbol{x})$ mod $\mathcal{P}(B)$, $\boldsymbol{u} \leftarrow \boldsymbol{u}' + \boldsymbol{x}$.
Return $(\boldsymbol{u}, \boldsymbol{u}')$.

Algorithm 4: The `NewPair` function

`Input` : A basis B, $R > 0$, a set $\mathcal{T} \subseteq L \times \mathcal{B}(R)$, $\gamma > 0$.
`Output` : A list of pairs $\mathcal{T}'$.
$\mathcal{C} \leftarrow \emptyset$, $\mathcal{T}' \leftarrow \emptyset$.
For each pair $(\boldsymbol{t}, \boldsymbol{t}') \in \mathcal{T}$ do
    If $\exists (\boldsymbol{c}, \boldsymbol{c}') \in \mathcal{C}, \|\boldsymbol{t}' - \boldsymbol{c}'\| \leq \gamma R$, then add $(\boldsymbol{t} - \boldsymbol{c}, \boldsymbol{t}' - \boldsymbol{c})$ to $\mathcal{T}'$,
    Else add $(\boldsymbol{t}, \boldsymbol{t}')$ to $\mathcal{C}$.
Return $\mathcal{T}'$

Algorithm 5: The `Sieve` algorithm

`Input` : A basis B, $\xi > \frac{1}{2}$, $\gamma > 0$, N, $\lambda \approx \lambda(L)$.
`Output` : A shortest non-zero vector of L.
$R \leftarrow n \cdot \max_i \|\boldsymbol{b}_i\| + \xi$, $\mathcal{T} \leftarrow \emptyset$.
For $i = 1$ to N do
    $\boldsymbol{x} \leftarrow$ random point uniformly chosen in $\mathcal{B}(\xi\lambda)$,
    Add `NewPair`$(B, \boldsymbol{x})$ to $\mathcal{T}$.
For $i = 1$ to $\left\lceil \log_\gamma \left(\frac{\xi}{nR(1-\gamma)} \right) \right\rceil$ do
    $\mathcal{T} \leftarrow$ `Sieve`$(B, R, \mathcal{T})$, $R \leftarrow \gamma R + \xi$.
Remove pairs of $\mathcal{T}$ corresponding to the same lattice point.
Return the difference between two closest distinct lattice points in $\mathcal{T}$ (fail if they do not exist).

Algorithm 6: The `AKS` algorithm

Theorem 4.2. *Let $(\boldsymbol{b}_i)_{i \leq n}$ be a basis of a lattice L such that $\max_i \|\boldsymbol{b}_i\| \leq 2^{3n}\lambda(L)$. Let $\xi = \frac{\sqrt{2}}{2}$, $\gamma = 2 - \sqrt{2}$, $N = 2^{2.173n}$ and $\lambda \in [\lambda(L), (1 + 1/n)\lambda(L))$. With probability exponentially close to 1,* `AKS` *returns a shortest vector of $L \setminus \mathbf{0}$. Moreover, it terminates in time $\leq \mathcal{P}oly(n)2^{3.346n}$ and space $\leq \mathcal{P}oly(n)2^{2.173n}$.*

It is possible to ensure the first assumption is fulfilled by LLL-reducing the basis B and removing vectors $\boldsymbol{b}_i$ such that $\|\boldsymbol{b}_i\| > 2^{2n}\|\boldsymbol{b}_1\|$. This does not change the shortest vectors (see [65, Le. 3.3]). Although $\lambda(L)$ is unknown, running the algorithm with $\lambda = \|\boldsymbol{b}_1\|(1 + 1/n)^{-i}$ for $i = 0, \ldots, \frac{n}{2\log_2\left(1+\frac{1}{n}\right)} = O(n^2)$ ensures that one value of λ will satisfy the condition on λ, by LLL-reducedness.

We describe the four main steps of the proof. More details are given in the appendix. For the whole proof, an arbitrary shortest vector $\boldsymbol{s}$ is fixed. Only the following subset of the sampled pairs is ever considered: the *good* pairs $(\boldsymbol{t}, \boldsymbol{t}')$ are those such that the perturbation $\boldsymbol{x} = \boldsymbol{t}' - \boldsymbol{t}$ belongs to $\mathcal{B}(\mathbf{0}, \xi\lambda) \cap \mathcal{B}(-\boldsymbol{s}, \xi\lambda)$. The first step consists in proving that a sampled pair is good with probability $\geq 2^{\frac{n}{2}\log_2\left(1-\frac{1}{4\xi^2}\right)+o(n)}$. This is done via elementary geometry arguments. The second step consists in bounding the number of vectors that are used as centers (and lost): There are $\leq 2^{(-\log_2\gamma+0.401)n+o(n)}$ centers. The third step consists in proving that, with high probability and at the end of the execution, the set $\mathcal{T}$ contains the same lattice vector twice. This is done by bounding the number of elements in $L \cap \mathcal{B}(r_0\lambda(L))$ which contains $\mathcal{T}$ by $2^{(\log_2 r_0+0.401)n+o(n))}$. The last two steps are based on the saturation phenomenon (Theorem 4.1). Finally, it can be shown that the probability of success is at least as large as the probability of collision (up to a constant factor). This implies that `AKS` succeeds with probability exponentially close to 1. Optimization on r_0 and ξ leads to the constants of Theorem 4.2.

Improving the complexity by using the birthday paradox. In the analysis of `AKS`, we use the fact that before removing identical lattice vectors in $\mathcal{T}$, the set contains two good pairs with the same lattice vector with high probability: This is because $\mathcal{T}$ contains $> N_B(n)$ lattice vectors corresponding to good pairs,

and $\mathcal{T}$ is itself contained in a set of cardinality $\leq N_B(n)$. If the vector were iid, no more than $O(\sqrt{N_B(n)})$ vectors would suffice. Although this may not hold for the version of AKS given above, the algorithm can be modified so that the iid-ness condition is fulfilled.

The change is as follows: At the start of the execution, we fix the set of vectors that will be used as centers at every step. Other vectors cannot be used as centers, so if at any sieving step, no close enough center is found, the vector under scope is lost. The probability of loss of each vector during the whole process can be made exponentially low by choosing the initial number of pairs carefully. These modifications can be implemented without incurring a significant cost increase (see the appendix for details). On the other hand, the number of vectors needed for the last step of the algorithm is decreased by an exponential factor. This leads to an algorithm in time $\leq 2^{2.571n}$ and space $\leq 2^{1.407n}$ when $\gamma = 0.5892$ and $\xi = 0.9365$.

Heuristic version of AKS. A heuristic version of AKS has been studied by Nguyen and Vidick [65]. The main modification consists in not using perturbations. To generate lattice vectors, random vectors are sampled using the randomized version of Babai's nearest plane algorithm [7] described in [45]. Under an assumption of uniform distribution of sieved points, they show that the space complexity of their algorithm is $\leq 2^{0.208n}$. In practice, the heuristic version solves SVP up to dimension 50. A refinement of this heuristic has been proposed by Wang *et al.* [83].

4.2 The ListSieve Algorithm

ListSieve was introduced by Micciancio and Voulgaris [58]. It can be described as a variant of AKS in which the order of the loop in the Sieve function and the second loop of AKS is reversed: in ListSieve, the outer loop is on vectors and the inner loop is on norm reduction steps. This makes the following improvement possible: for a given point, instead of using separate sets of centers for each step, all centers are put together in the same list. At a given step, any item in the list may be used provided that it decreases the norm of the vector by a factor $\geq 1/\gamma$. The loss of ξ between each step $R \leftarrow \gamma R + \xi$ of AKS does not occur in ListSieve, making it possible to set γ polynomially close to 1 (e.g., $\gamma = 1 - 1/n$) to keep the list size as small as possible.

```
Input  : A pair (u, u') and a list T ⊆ L.
Output : A reduced pair (u, u').
While ∃w ∈ T, ‖u' − w‖ < γ‖u'‖ do
    (u, u') ← (u − w, u' − w).
Return (u, u').
```

Algorithm 7: The Reduction algorithm

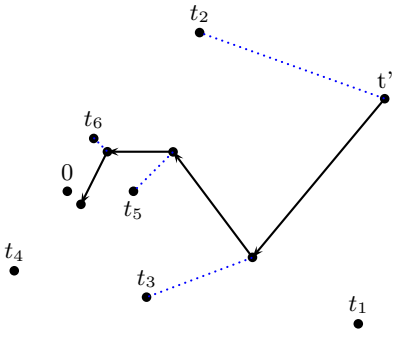


Fig. 3. Reduction of $\boldsymbol{t}'$ with the list $\{\boldsymbol{t}_1, \ldots, \boldsymbol{t}_6\}$

```
Input  : A basis B, λ ≈ λ(L), ξ > 1/2, N.
Output : A shortest non-zero vector of L.
Choose (x_1, ..., x_N) independently and uniformly in B(ξλ). T ← {0}.
For i = 1 to N do
    (t_i, t'_i) ← Reduction(NewPair(B, x_i), T),
    Add t_i to T unless it already belongs to it.
Find closest vectors (s_1, s_2) in T (fail if |T| = 1).
Return s_1 − s_2.
```

Algorithm 8: The `ListSieve` algorithm

```
Input  : A basis B, λ ≈ λ(L), ξ > 1/2, r_0 > 2ξ, N_1, N_2.
Output : A shortest non-zero vector of L.
Choose (x_1, ..., x_{N_1}, y_1, ..., y_{N_2}) iid uniform in B(ξλ). T ← ∅, U ← ∅.
For i = 1 to N_1 do
    (t_i, t'_i) ← Reduction(NewPair(B, x_i), T).
    If ||t_i|| ≥ r_0 λ then Add t_i to T unless it already belongs to it.
For i = 1 to N_2 do
    (u_i, u'_i) ← Reduction(NewPair(B, y_i), T),
    Add t_i to U unless it already belongs to it.
Find closest vectors (s_1, s_2) in U (fail if |U| = 1).
Return s_1 − s_2.
```

Algorithm 9: The `ListSieveBirthday` algorithm

Theorem 4.3. *Let $(\boldsymbol{b}_i)_{i\leq n}$ be a basis of a lattice L. If $\lambda \in [\lambda(L), (1+1/n)\lambda(L))$, $N = 2^{1.874n}$, $\xi = 0.685$ and n is sufficiently large, then* `ListSieve` *returns a shortest non-zero vector of L with probability $\geq \frac{1}{3}$, in time $\leq \mathcal{P}oly(n)2^{3.199n}$ and space $\leq \mathcal{P}oly(n)2^{1.325n}$.*

The proof of `ListSieve` is similar to that of `AKS`. The size of the list is bounded with Theorem 4.1, using two properties: any vector is reduced with respect to the previous vectors of the list and all vectors in the list belong to the lattice. Because

of this, collisions occur when enough pairs are sampled. As in the proof of AKS, it can be proved that the probability of success is nearly as high as the probability that a collision occurs with a *good* vector. A detailed proof is given in appendix.

Improving the complexity by using the birthday paradox. In the original version of ListSieve, the vectors of the list may not be statistically independent. Dividing the algorithm into two phases makes it possible to sample iid vectors in a small ball $\mathcal{B}(r_0\lambda)$ (see Algorithm 9). In this version, the birthday paradox can be used to analyze the second phase, which leads to smaller time and space complexity bounds than for all versions of AKS.

Theorem 4.4. *Let $(\boldsymbol{b}_i)_{i\leq n}$ be a basis of a lattice L. If $\lambda \in [\lambda(L), (1+1/n)\lambda(L))$, $\xi = 0.9476$, $r_0 = 3.0169$, N_1 chosen uniformly in $[0, \lfloor 2^{1.3501n} \rfloor]$, $N_2 = 2^{1.2325n}$ and n is sufficiently large, then* ListSieveBirthday *returns a shortest non-zero vector of L with probability exponentially close to 1, in time $\leq \mathcal{P}oly(n)2^{2.465n}$ and space $\leq \mathcal{P}oly(n)2^{1.233n}$.*

During the first phase of the algorithm, very short vectors are thrown away. This allows to improves the bound on $|\mathcal{T}|$ from ListSieve. Indeed, because of perturbations, the lower bound for the angle between two vectors is worse for vectors of small norm. During the second phase, lattice vectors are reduced with respect to the list of vectors obtained during the first phase. As long as $|\mathcal{T}|$ is large enough, this should produce short lattice vectors (e.g., vectors in $\mathcal{B}(r_0\lambda)$). Unfortunately, it is unclear whether the probability for a vector to be short decreases when $|\mathcal{T}|$ increases. This is why $|\mathcal{T}|$ is randomized. The proof that the difference between two vectors in $\mathcal{U}$ is a shortest non-zero lattice vector is the same as for AKS. More details are given in appendix.

Heuristic version of ListSieve. In [58], Micciancio and Voulgaris give experimental results on a heuristic version of ListSieve that they call GaussSieve. In GaussSieve, vectors are more reduced with respect to each other. As in the heuristic version of AKS, there are no perturbations: It is an open problem whether there is a way to guarantee that the algorithm succeeds. Thus, the stopping condition is chosen heuristically. However, a space complexity upper bound of $2^{0.401n}$ is proved (this corresponds to the best known upper bound on the so-called kissing number). According to the experiments from [58], GaussSieve can be used up to dimension 63 and outperforms the (guaranteed) deterministic enumeration algorithm from Section 5 without tree pruning.

4.3 Solving CVP

In 2002, Ajtai *et al.* [6] showed how to modify their SVP solver to solve CVP within any fixed factor $1+\varepsilon$ for $\varepsilon > 0$, in time and space $2^{O(n)}$. In [9], Blömer and Naewe showed that the sieve algorithms can be used to solve $(1+\varepsilon)$-CVP by reducing the latter to the "Generalized SVP" $(1+\varepsilon)$-GSVP, which, given a basis of a lattice L and a linear subspace M of $\mathbb{R}^m$, consists in finding a vector

in $L \setminus M$ of norm $\leq (1+\varepsilon)\min_{\|\cdot\|}(L \setminus M)$. They proved that $(1+\varepsilon)$-GSVP can be solved by `AKS` just by changing the parameters and the last step, in which a condition is added: The difference between the two vectors must not be in M. As the shortest vector outside M can be much larger than $\lambda(L)$, it may not be possible to "fill" the ball $\mathcal{B}(r_0\lambda)$ corresponding to the end of the execution of `AKS`. The analysis of `AKS` does not carry over, but it is proved instead that `AKS` returns a vector outside of M of norm at most $r_0\lambda$. The approximation factor $1+\varepsilon$ is obtained by choosing γ and ξ so that $r_0 \leq 1+\varepsilon$. This is possible for any $\varepsilon > 0$, but at the expense of increasing the complexity bound (the exponent is polynomial in $\frac{1}{\varepsilon}$). The reduction from $(1+\varepsilon)$-CVP to $(1+\varepsilon)$-GSVP is based on Kannan's embedding technique [43, Se. 6]: For a lattice $L \subseteq \mathbb{R}^m$ and a target $\boldsymbol{t} \in \mathbb{R}^m$, we define L' as the lattice spanned by $(L \times \{0\}) \cup (\boldsymbol{t}, \gamma)$; provided that γ is large enough, if $(\boldsymbol{u}, x) \in \mathbb{R}^m \times \mathbb{R}$ is a shortest vector of $L' \setminus (\mathbb{R}^n \times \{0\})$, then $x = \pm\gamma$ and the solution to $\mathrm{CVP}(L, \boldsymbol{t})$ is $\pm\boldsymbol{u}$. More recently, Eisenbrand *et al.* [18] built upon [9] to decrease the cost dependence in ε.

5 Enumeration-Based Solvers

The oldest SVP/CVP solvers rely on an enumeration process, that, given a basis $(\boldsymbol{b}_i)_{i\leq n}$ of lattice L, a center $\boldsymbol{t}$ and a radius A, looks for all points of $L \cap \mathcal{B}(\boldsymbol{t}, A)$. It does so by enumerating all points of projections of L orthogonally to basis vectors, that belong to hyperballs of smaller dimensions. In practice, a heuristic version of the enumeration based on pruning has been used to solve SVP for a generic-looking lattice of dimension 110, within a few days of computation [25,26].

The main ingredient for the complexity analyzes of enumeration-based solvers consists in bounding and/or estimating the number of lattice points within a ball. Estimates are provided by the so-called *Gaussian heuristic*: if K is a measurable subset of the span of the n-dimensional lattice L, then $|K \cap L| \approx \mathrm{vol}(K)/\det(L)$ (where vol denotes the n-dimensional volume). For some choices of K compact and sufficiently large, or K or L sampled randomly, then rigorous versions of the Gaussian heuristic can be obtained. We will use the Gaussian heuristic mainly for balls, in which case we have $\mathrm{vol}\,\mathcal{B}_n(\boldsymbol{t}, A) = \frac{A^n \pi^{n/2}}{\Gamma(n/2+1)} \approx \frac{2^{O(n)} A^n}{n^{n/2}}$, for any $\boldsymbol{t}$ and A.

5.1 The `Enum` Algorithm

`Enum` (Algorithm 10) enumerates $L \cap \mathcal{B}(\boldsymbol{t}, A)$ by using the triangular relationship between the basis $(\boldsymbol{b}_i)_{i\leq n}$ of L and its Gram-Schmidt orthogonalization $(\boldsymbol{b}_i^*)_{i\leq n}$. More specifically, it relies on the two following observations:

- If $\boldsymbol{x} = \sum_i x_i \boldsymbol{b}_i$ belongs to $L \cap \mathcal{B}(\boldsymbol{t}, A)$, then, for any $i \leq n$, we have $\boldsymbol{x}^{(i)} \in L^{(i)} \cap \mathcal{B}(\boldsymbol{t}^{(i)}, A)$, where $\boldsymbol{x}^{(i)}$, $L^{(i)}$ and $\boldsymbol{t}^{(i)}$ are the projections of $\boldsymbol{x}$, L and $\boldsymbol{t}$ orthogonally to the linear span of $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{i-1}$.

- Enumerating $L^{(n)} \cap \mathcal{B}(\boldsymbol{t}^{(n)}, A)$ is easy and once $L^{(i+1)} \cap \mathcal{B}(\boldsymbol{t}^{(i+1)}, A)$ is known, it is easy to enumerate $L^{(i)} \cap \mathcal{B}(\boldsymbol{t}^{(i)}, A)$. Indeed: Assume that $\boldsymbol{x}^{(i)} \in L^{(i)} \cap \mathcal{B}(\boldsymbol{t}^{(i)}, A)$; Write $\boldsymbol{x}^{(i)} = \boldsymbol{x}^{(i+1)} + (x_i + c_i)\boldsymbol{b}_i^*$ for some $x_i \in \mathbb{Z}$ and $c_i \in \mathbb{R}$. Once $\boldsymbol{x}^{(i+1)} \in L^{(i+1)} \cap \mathcal{B}(\boldsymbol{t}^{(i+1)}, A)$ is fixed, we must have

$$x_i \in \mathbb{Z} \cap \left[-c_i - \frac{\sqrt{A^2 - \|\boldsymbol{x}^{(i+1)}\|^2}}{\|\boldsymbol{b}_i^*\|}, -c_i + \frac{\sqrt{A^2 - \|\boldsymbol{x}^{(i+1)}\|^2}}{\|\boldsymbol{b}_i^*\|} \right] \tag{2}$$

During its execution, `Enum` considers all points in $L^{(i)} \cap \mathcal{B}(\boldsymbol{t}^{(i)}, A)$, for $i = n, n-1, \ldots, 1$. An inherent drawback is that the complexity may be (significantly) more than $|L \cap \mathcal{B}(\boldsymbol{t}, A)|$. This is because it often happens that at some stage, an element of $L^{(i+1)} \cap \mathcal{B}(\boldsymbol{t}^{(i+1)}, A)$ has no descendant in $L^{(i)} \cap \mathcal{B}(\boldsymbol{t}^{(i)}, A)$ (i.e., the interval in (2) contains no integer) : This corresponds to a "dead-end" in the enumeration tree.

`Input` : A basis $(\mathbf{b}_i)_{i \le n}$ of a lattice L, $\boldsymbol{t} \in \mathrm{span}(L)$, $A > 0$.
`Output` : All vectors in $L \cap \mathcal{B}(\boldsymbol{t}, A)$.
Compute the $\mu_{i,j}$'s and $\|\boldsymbol{b}_i^*\|^2$'s.
Compute the t_i's such that $\mathbf{t} = \sum_i t_i \mathbf{b}_i^*$.
$S \leftarrow \emptyset, \boldsymbol{\ell} \leftarrow \mathbf{0}, \boldsymbol{x} \leftarrow \mathbf{0}$, $x_n \leftarrow \lceil t_n - A/\|\mathbf{b}_n^*\| \rceil$, $i \leftarrow n$.
While $i \le n$ do
  $\ell_i \leftarrow (x_i - t_i + \sum_{j>i} x_j \mu_{j,i})^2 \|\mathbf{b}_i^*\|^2$,
  If $i = 1$ and $\sum_{1 \le j \le n} \ell_j \le A^2$ then
    $S \leftarrow S \cup \{\mathbf{x}\}$, $x_1 \leftarrow x_1 + 1$.
  If $i \ne 1$ and $\sum_{j \ge i} \ell_j \le A^2$ then
    $i \leftarrow i - 1$, $x_i \leftarrow \left\lceil t_i - \sum_{j>i} (x_j \mu_{j,i}) - \sqrt{\frac{A^2 - \sum_{j>i} \ell_j}{\|\mathbf{b}_i^*\|^2}} \right\rceil$.
  If $\sum_{j \ge i} \ell_j > A$, then $i \leftarrow i + 1$, $x_i \leftarrow x_i + 1$.
Return S.

Algorithm 10: The `Enum` algorithm

The cost of `Enum` can be bounded by $\sum_i |L^{(i)} \cap \mathcal{B}(\boldsymbol{t}^{(i)}, A)|$, up to a small polynomial factor. The Gaussian heuristic leads to the approximation (for $i \le n$):

$$|L^{(i)} \cap \mathcal{B}(\boldsymbol{t}^{(i)}, A)| \approx \frac{2^{O(n)} A^{n-i+1}}{(n-i+1)^{\frac{n-i+1}{2}} \cdot \prod_{j=i}^{n} \|\boldsymbol{b}_j^*\|}.$$

Unfortunately, some of the involved balls are very small compared to their corresponding lattice $L^{(i)}$, and it seems hard to prove that the heuristic always holds. Though of mostly theoretical nature (because of the fuzzy $2^{O(n)}$ factor), the following result provides theoretical evidence towards the validity of the Gaussian heuristic in the present situation.

Theorem 5.1 ([32]). *The bit-complexity of* `Enum` *can be bounded from above by:*

$$2^{O(n)} \prod_{1 \le i \le n} \max\left(1, \frac{A}{\sqrt{n}\|\boldsymbol{b}_i^*\|}\right) \le 2^{O(n)} \max_{I \subseteq [1,n]} \frac{A^{|I|}}{\sqrt{n}^{|I|} \cdot \prod_{i \in I} \|\boldsymbol{b}_i^*\|}.$$

`Enum` may be used directly to solve SVP and CVP, once the bound A has been set. In the case of SVP, it may be derived from Minkowski's theorem, or from the current basis $(\mathbf{b}_i)_{i \leq n}$: For example, one may choose $A = \min(\min_i \|\mathbf{b}_i\|, \sqrt{n}(\det L)^{1/n})$. In the case of CVP, it may be derived from any bound on $\mu(L)$, such as $\sqrt{\sum_i \|\boldsymbol{b}_i^*\|^2}/2$. The bound may also be set heuristically using the Gaussian heuristic: The guess for A is then derived from $\operatorname{vol} \mathcal{B}(\boldsymbol{t}, A) \approx \det L$, and is increased if no solution is found. The bound A can also be decreased during the execution of `Enum`, every time a better solution is found, as the complexity increases sharply with A.

As written, the space required by `Enum` may be more than $\mathcal{P}oly(n)$. It can be made $\mathcal{P}oly(n)$ for the SVP and CVP applications, as only a single shortest/closest vector is required: the update of S in `Enum` should then be replaced by an update of the best solution found so far.

5.2 Reducing before Enumerating

The cost estimates and upper bounds of `Enum` strongly depend on A and on the decrease of the $\|\boldsymbol{b}_i^*\|$'s. This observation suggests that the more reduced the basis $(\boldsymbol{b}_i)_i$, the lower the cost. Fincke and Pohst [20] initially used a LLL-reduced basis $(\boldsymbol{b}_i)_i$. For such a basis, we have $\|\boldsymbol{b}_{i+1}^*\| \geq \|\boldsymbol{b}_i^*\|/2$, which leads to a $2^{O(n^2)}$ complexity upper bound. Kannan [42] observed that the cost of `Enum` is so high that a much more aggressive pre-processing significantly lowers the total cost while negligibly contributing to it. In practice, one may use BKZ: This still leads to a $2^{O(n^2)}$ complexity bound for `Enum` when the block-size is fixed, but the $O(\cdot)$ constant decreases as the inverse of the block-size (up to factors of lower order). Kannan's theoretical algorithms are even more aggressive than that. His SVP algorithm is in fact an HKZ-reduction algorithm that calls itself recursively in lower dimensions to optimize the reduction before the calls to `Enum`.

The HKZ-reduction algorithm proceeds as follows (we refer to [36] for more details). Before calling `Enum` to find a shortest non-zero lattice vector, it quasi-HKZ-reduces the $\boldsymbol{b}_i$'s: at the moment `Enum` is called, the basis is such that once projected orthogonally to $\boldsymbol{b}_1$ it is HKZ-reduced, and $\|\boldsymbol{b}_2^*\| \geq \|\boldsymbol{b}_1\|/2$. To find such a basis, it suffices to:

- HKZ-reduce (in dimension $n-1$) the projections of the $\boldsymbol{b}_i$'s orthogonally to $\boldsymbol{b}_1$;
- extend them to a basis $(\boldsymbol{b}_i)_i$ of L, while maintaining that property and keeping the previous $\boldsymbol{b}_1$;
- HKZ-reduce $(\boldsymbol{b}_1, \boldsymbol{b}_2)$ (and update the basis of L accordingly);
- and iterate the previous steps until quasi-HKZ-reduction is reached.

Then a shortest vector $\boldsymbol{b}$ of $L \setminus \mathbf{0}$ can be found with `Enum`, and extended into a basis of L (keeping $\boldsymbol{b}$ in first position). The HKZ-reduction algorithm is then called recursively on the projection of the basis orthogonally to $\boldsymbol{b}$, to get a reduced basis of L.

A detailed analysis of quasi-HKZ-reduced bases $(\boldsymbol{b}_i)_{i\leq n}$ gives that (see [32]):

$$\max_{I\subseteq[1,n]} \frac{\|\boldsymbol{b}_1\|^{|I|}}{\sqrt{n}^{|I|}\cdot\prod_{i\in I}\|\boldsymbol{b}_i^*\|} \leq 2^{O(n)} n^{n/(2e)}.$$

The call to `Enum` dominates the overall cost of the HKZ-reduction algorithm, so that Kannan's SVP solver terminates within $n^{n/(2e)+o(n)}$ bit operations.

Kannan's CVP algorithm is `Enum`, starting with an HKZ-reduced basis $(\boldsymbol{b}_i)_{i\leq n}$. Wlog, one may assume that $\|\boldsymbol{b}_1\| = \max_i \|\boldsymbol{b}_i\|$ (see [42]), and take $A = \sqrt{n}\|\boldsymbol{b}_1\|$. In that situation, we have:

$$\max_{I\subseteq[1,n]} \frac{(\sqrt{n}\|\boldsymbol{b}_1\|)^{|I|}}{\sqrt{n}^{|I|}\prod_{i\in I}\|\boldsymbol{b}_i^*\|} = \frac{\|\boldsymbol{b}_1\|^n}{\det L} \leq n^{n/2},$$

by Minkowski's theorem. This leads to an overall $n^{n/2+o(n)}$ complexity upper bound for Kannan's CVP solver.

5.3 Practical Improvements

As `Enum` is the most practical SVP/CVP solver, much effort has been spent on optimizing it. Schnorr and Euchner [75] introduced a zig-zag strategy for enumerating the interval of Equation (2): starting from the middle of the interval increases the likeliness of finding short/close vectors faster, and to decrease A quickly.

Recently, Gama *et al.* [25, App. D] described a way of computing the term $\sum_{j>i} x_j\mu_{j,i}$ from `Enum` more efficiently by storing partial sums that can be re-used later in the execution, and thus removing redundant operations. This can save a significant amount of computation, at the expense of a small increase in space requirement.

In `Enum`, the quantities $(\mu_{i,j}, \|\boldsymbol{b}_i^*\|^2, t_i, A)$ are rational numbers, so that all computations can be performed exactly. However, the bitsizes of these numbers can be as large as $O(n\log B)$ (where B is the largest bitsize of an entry in the input basis matrix), which leads to a large arithmetic cost. In practice, approximate arithmetics such as floating-point or fixed-point are used instead. Analyzing the effect of these approximate computations requires to undertake a detailed analysis of the accumulation and amplification of the errors which occur at every step, as they could lead to incorrect results. These issues have been studied in detail in [67,16]. It is possible to derive a sufficient precision such that when starting with an arbitrary LLL-reduced basis, the output is guaranteed to be a close to optimal solution. However, a dynamic analysis of the error should be preferred, because the bounds on diverse quantities used in the worst-case analysis are very pessimistic.

The parallelization of `Enum` has been investigated in [37] and [16]. In the latter, the authors use the Gaussian heuristic to balance the sizes of the sub-tasks sent to the slave machines.

The main heuristic used for accelerating `Enum` consists in pruning the corresponding enumeration tree, by cutting off branches with low ratio between the estimated branch size and the likeliness of containing the desired solution [75,76,80,25]. Pruning consists in replacing the sets $L^{(i)} \cap \mathcal{B}(\boldsymbol{t}^{(i)}, A)$ by subsets $L^{(i)} \cap \mathcal{B}(\boldsymbol{t}^{(i)}, p_i \cdot A)$, for some pruning coefficients $0 < p_n \leq \ldots \leq p_1 \leq 1$. The approximate branch size and success likeliness can be derived from the Gaussian heuristic, by estimating the volumes of the following truncated hyperballs (for all $i \leq n$):

$$\left\{(y_j)_{i \leq j \leq n} \in \mathbb{R}^{n-i+1} : \forall k \geq j, \left\|(y_j)_{k \leq j \leq n} - \mathbf{t}^{(k)}\right\| \leq p_k \cdot A\right\}.$$

Gama *et al.* recently proposed in [25] a further refinement, called extreme pruning. It consists in calling an enumeration with aggressive pruning coefficients several times on randomized inputs: this allows the success probability of a single call to be lowered, and to decrease the overall cost even more.

6 Open Problems

Many important techniques and results on solving SVP and CVP have been discovered in the last few years. The recent increase of interest in this topic stems at least in large part from the rise of lattice-based cryptography. We draw here a list of open problems that we find of particular importance.

The description and analysis of the Voronoi-based solver from Section 3 is extremely recent [57]. So far, no practical implementation nor heuristic improvement that could lead to a competitive implementation is known. Is the worst-case complexity bound sharp, and if so, is it likely to be reached for "average" inputs? Are there reasonable heuristics that could be exploited to speed up this algorithm?

The solvers relying on saturation and enumeration have undergone significantly more scrutiny, but some important questions remain open. For example, it is unclear whether the perturbations of the lattice vectors in saturation-based solvers are inherently necessary or just an artifact of the proof. As these perturbations lead to increased complexity bounds, proving them unnecessary could make these solvers competitive with [57]. Also, is it a valid heuristic to remove them in practice?

The theoretical and practical efficiency of enumeration-based solvers greatly depends on the pre-processing of the input basis. It is suggested in [62] that enumeration-based SVP solvers cost at least $n^{n/8+o(n)}$ in the worst case: Is it possible to reach this lower bound? In practice, what is the best trade-off between pre-processing and enumerating? With respect to enumeration-based CVP solvers, an intriguing question is whether the best known complexity upper bound $n^{n/2+o(n)}$ can be reached in the worst case: contrarily to SVP [33], there is a gap between the best known worst-case complexity upper and lower bounds.

Overall, we have three very different families of solvers for SVP/CVP, which seem largely unrelated (except the dimension reduction step in the algorithm

based on the Voronoi cell, which is inspired from `Enum`). Furthermore, their complexity upper bounds are incomparable: the solvers relying on the Voronoi cell and on saturation have time and space complexity upper bounds that are simply exponential, whereas Kannan's SVP solver has a worse time complexity bound but requires polynomially bounded space. Are there hidden links between these algorithms? Can they be combined to achieve interesting time/space trade-offs? Enumeration-based heuristic solvers are currently more practical for handleable dimensions, but the other solvers enjoy lower asymptotic complexity bounds: is it possible to estimate the cut-off dimension after which they become the most efficient?

It is also conceivable that faster solvers exist, that remain to be discovered. For example, is it possible to achieve exponential time complexity with a polynomially bounded space requirement? Are there ways to exploit quantum computations to obtain better complexity bounds? Finally, lattice-based cryptography does not rely on the hardness of SVP but of relaxed variants of it. In particular, if there were a polynomial-time algorithm that could find non-zero lattice vectors that are no more than polynomially longer (in the dimension) than the lattice minimum, then lattice-based cryptography would become insecure. Does such an SVP solver exist, or is it possible to argue that it does not exist?

Acknowledgments. We thank Panagiotis Voulgaris for very helpful discussions on the Voronoi-based SVP/CVP solver. We also thank the anonymous reviewer for her/his comments. The third author was partly supported by the Australian Research Council under ARC Discovery Grant DP110100628.

References

1. Agrell, E., Eriksson, T., Vardy, A., Zeger, K.: Closest point search in lattices. IEEE Transactions on Information Theory 48(8), 2201–2214 (2002)
2. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: Proc. of STOC, pp. 99–108. ACM, New York (1996)
3. Ajtai, M.: The worst-case behavior of Schnorr's algorithm approximating the shortest nonzero vector in a lattice. In: Proc. of STOC, pp. 396–406. ACM, New York (2003)
4. Ajtai, M., Dwork, C.: A public-key cryptosystem with worst-case/average-case equivalence. In: Proc. of STOC, pp. 284–293. ACM, New York (1997)
5. Ajtai, M., Kumar, R., Sivakumar, D.: A sieve algorithm for the shortest lattice vector problem. In: Proc. of STOC, pp. 601–610. ACM, New York (2001)
6. Ajtai, M., Kumar, R., Sivakumar, D.: Sampling short lattice vectors and the closest lattice vector problem. In: Proc. of CCC, pp. 53–57 (2002)
7. Babai, L.: On Lovász lattice reduction and the nearest lattice point problem. Combinatorica 6, 1–13 (1986)
8. Banaszczyk, W.: New bounds in some transference theorems in the geometry of numbers. Math. Ann. 296, 625–635 (1993)
9. Blömer, J., Naewe, S.: Sampling methods for shortest vectors, closest vectors and successive minima. Theor. Comput. Science 410(18), 1648–1665 (2009)

10. Bosma, W., Cannon, J., Playoust, C.: The Magma algebra system. I. The user language. Journal of Symbolic Computation 24(3-4), 235–265 (1997), http://magma.maths.usyd.edu.au/magma/
11. Buchmann, J.: Reducing Lattice Bases by Means of Approximations. In: Huang, M.-D.A., Adleman, L.M. (eds.) ANTS 1994. LNCS, vol. 877, pp. 160–168. Springer, Heidelberg (1994)
12. Cadé, D., Pujol, X., Stehlé, D.: fplll-3.1, a floating-point LLL implementation, http://perso.ens-lyon.fr/damien.stehle
13. Cassels, J.W.S.: An Introduction to the Geometry of Numbers, 2nd edn. Springer, Heidelberg (1971)
14. Conway, J.H., Sloane, N.J.A.: Sphere Packings, Lattices and Groups, 3rd edn. Springer, Heidelberg (1998)
15. Dadush, D., Peikert, C., Vempala, S.: Enumerative algorithms for the shortest and closest lattice vector problems in any norm via M-ellipsoid coverings (submitted 2011)
16. Detrey, J., Hanrot, G., Pujol, X., Stehlé, D.: Accelerating lattice reduction with fPGAs. In: Abdalla, M., Barreto, P.S.L.M. (eds.) LATINCRYPT 2010. LNCS, vol. 6212, pp. 124–143. Springer, Heidelberg (2010)
17. Eisenbrand, F.: Integer Programming and Algorithmic Geometry of Numbers. In: 50 Years of Integer Programming 1958-2008, From the Early Years to the State-of-the-Art. Springer, Heidelberg (2009)
18. Eisenbrand, F., Hähnle, N., Niemeier, M.: Covering cubes and the closest vector problem. To appear in the Proceedings of SoCG (2011)
19. van Emde Boas, P.: Another NP-complete partition problem and the complexity of computing short vectors in a lattice. Technical report 81-04, Mathematisch Instituut, Universiteit van Amsterdam (1981)
20. Fincke, U., Pohst, M.: A procedure for determining algebraic integers of given norm. In: van Hulzen, J.A. (ed.) ISSAC 1983 and EUROCAL 1983. LNCS, vol. 162, pp. 194–202. Springer, Heidelberg (1983)
21. Fincke, U., Pohst, M.: Improved methods for calculating vectors of short length in a lattice, including a complexity analysis. Math. Comp. 44(170), 463–471 (1985)
22. Gama, N., Howgrave-Graham, N., Koy, H., Nguyên, P.Q.: Rankin's constant and blockwise lattice reduction. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 112–130. Springer, Heidelberg (2006)
23. Gama, N., Nguyen, P.Q.: Finding short lattice vectors within Mordell's inequality. In: Proc. of STOC, pp. 207–216. ACM, New York (2008)
24. Gama, N., Nguyen, P.Q.: Predicting lattice reduction. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 31–51. Springer, Heidelberg (2008)
25. Gama, N., Nguyen, P.Q., Regev, O.: Lattice enumeration using extreme pruning. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 257–278. Springer, Heidelberg (2010)
26. Gama, N., Schneider, M.: The SVP challenge homepage, http://www.latticechallenge.org/svp-challenge/
27. Goldreich, O., Goldwasser, S., Halevi, S.: Public-key cryptosystems from lattice reduction problems. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 112–131. Springer, Heidelberg (1997)
28. Goldreich, O., Micciancio, D., Safra, S., Seifert, J.-P.: Approximating shortest lattice vectors is not harder than approximating closest lattice vectors. Inf. Process. Lett. 71(2), 55–61 (1999)
29. Goldstein, D., Mayer, A.: On the equidistribution of Hecke points. Forum Mathematicum 15, 165–189 (2003)

30. Gruber, M., Lekkerkerker, C.G.: Geometry of Numbers. North-Holland, Amsterdam (1987)
31. Guruswami, V., Micciancio, D., Regev, O.: The complexity of the covering radius problem. Computational Complexity 14(2), 90–121 (2005)
32. Hanrot, G., Stehlé, D.: Improved analysis of kannan's shortest lattice vector algorithm (extended abstract). In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 170–186. Springer, Heidelberg (2007)
33. Hanrot, G., Stehlé, D.: Worst-case Hermite-Korkine-Zolotarev reduced lattice bases. CoRR, abs/0801.3331 (2008)
34. Hassibi, A., Boyd, S.: Integer parameter estimation in linear models with applications to GPS. IEEE Transactions on Signal Process 46(11), 2938–2952 (1998)
35. Haviv, I., Regev, O.: Tensor-based hardness of the shortest vector problem to within almost polynomial factors. In: Proc. of STOC, pp. 469–477. ACM, New York (2007)
36. Helfrich, B.: Algorithms to construct Minkowski reduced and Hermite reduced lattice bases. Theor. Comput. Science 41, 125–139 (1985)
37. Hermans, J., Schneider, M., Buchmann, J., Vercauteren, F., Preneel, B.: Parallel shortest lattice vector enumeration on graphics cards. In: Bernstein, D.J., Lange, T. (eds.) AFRICACRYPT 2010. LNCS, vol. 6055, pp. 52–68. Springer, Heidelberg (2010)
38. Hermite, C.: Œuvres. Gauthiers-Villars (1905)
39. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: A ring-based public key cryptosystem. In: Buhler, J.P. (ed.) ANTS 1998. LNCS, vol. 1423, pp. 267–288. Springer, Heidelberg (1998)
40. Horváth, Á.G.: On the Dirichlet-Voronoi cells of the unimodular lattices. Geometricæ Dedicata 63, 183–191 (1996)
41. Kabatyansky, G.A., Levenshtein, V.I.: Bounds for packings on a sphere and in space. Probl. Peredachi Inf. 14(1), 3–25 (1978)
42. Kannan, R.: Improved algorithms for integer programming and related lattice problems. In: Proc. of STOC, pp. 99–108. ACM, New York (1983)
43. Kannan, R.: Minkowski's convex body theorem and integer programming. Math. Oper. Res. 12(3), 415–440 (1987)
44. Khot, S.: Inapproximability results for computational problems on lattices. Chapter of [64]
45. Klein, P.N.: Finding the closest lattice vector when it's unusually close. In: Proc. of SODA, pp. 937–941. ACM, New York (2000)
46. Korkine, A., Zolotarev, G.: Sur les formes quadratiques. Math. Ann. 6, 336–389 (1873)
47. Lenstra, A.K., Lenstra Jr., H.W., Lovász, L.: Factoring polynomials with rational coefficients. Math. Ann. 261, 515–534 (1982)
48. Lenstra Jr., H.: Lattices. In: Buhler, J.P., Stevenhagen, P. (eds.) Algorithmic Number Theory, pp. 127–181. MSRI Publications, Cambridge University Press (2008)
49. Lindner, R., Rückert, M.: The lattice challenge homepage, `http://www.latticechallenge.org/`
50. Liu, Y.-K., Lyubashevsky, V., Micciancio, D.: On bounded distance decoding for general lattices. In: Díaz, J., Jansen, K., Rolim, J.D.P., Zwick, U. (eds.) APPROX 2006 and RANDOM 2006. LNCS, vol. 4110, pp. 450–461. Springer, Heidelberg (2006)
51. Lyubashevsky, V., Micciancio, D.: On bounded distance decoding, unique shortest vectors, and the minimum distance problem. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 577–594. Springer, Heidelberg (2009)

52. Martinet, J.: Perfect Lattices in Euclidean Spaces. Springer, Heidelberg (2002)
53. Micciancio, D.: Efficient reductions among lattice problems. In: Proc. of SODA, pp. 84–93. SIAM, Philadelphia (2008)
54. Micciancio, D., Goldwasser, S.: Complexity of lattice problems: a cryptographic perspective. Kluwer Academic Press, Dordrecht (2002)
55. Micciancio, D., Regev, O.: Lattice-based cryptography. In: Bernstein, D.J., Buchmann, J., Dahmen, E. (eds.) Post-Quantum Cryptography, pp. 147–191. Springer, Heidelberg (2009)
56. Micciancio, D., Voulgaris, P.: A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations, Draft of the full version of [57] (December 8, 2010), http://cseweb.ucsd.edu/~pvoulgar/pub.html
57. Micciancio, D., Voulgaris, P.: A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations. In: Proc. of STOC, pp. 351–358. ACM, New York (2010)
58. Micciancio, D., Voulgaris, P.: Faster exponential time algorithms for the shortest vector problem. In: Proc. of SODA, ACM, New York (2010)
59. Minkowski, H.: Geometrie der Zahlen. Teubner-Verlag, Stuttgart (1896)
60. Mow, W.H.: Maximum likelihood sequence estimation from the lattice viewpoint. IEEE Transactions on Information Theory 40, 1591–1600 (1994)
61. Mow, W.H.: Universal lattice decoding: Principle and recent advances. Wireless Communications and Mobile Computing, Special Issue on Coding and Its Applications in Wireless CDMA Systems 3(5), 553–569 (2003)
62. P. Q. Nguyen. Hermite's constant and lattice algorithms. Chapter of [64].
63. Nguyên, P.Q., Stehlé, D.: LLL on the average. In: Hess, F., Pauli, S., Pohst, M. (eds.) ANTS 2006. LNCS, vol. 4076, pp. 238–256. Springer, Heidelberg (2006)
64. Nguyen, P.Q., Vallée, B. (eds.): The LLL Algorithm: Survey and Applications. Information Security and Cryptography. Springer, Heidelberg (2009)
65. Nguyen, P.Q., Vidick, T.: Sieve algorithms for the shortest vector problem are practical. Journal of Mathematical Cryptology 2(2) (2008)
66. Odlyzko, A.M.: The rise and fall of knapsack cryptosystems. In: Cryptology and Computational Number Theory. Proc. of Symposia in Applied Mathematics, vol. 42, pp. 75–88. A.M.S, Providence (1990)
67. Pujol, X., Stehlé, D.: Rigorous and efficient short lattice vectors enumeration. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 390–405. Springer, Heidelberg (2008)
68. Pujol, X., Stehlé, D.: Solving the shortest lattice vector problem in time $2^{2.465n}$. Cryptology ePrint Archive (2009), http://eprint.iacr.org/2009/605
69. Regev, O.: Lecture notes of lattices in computer science, taught at the Computer Science Tel Aviv University, http://www.cs.tau.il/~odedr
70. O. Regev. On the complexity of lattice problems with polynomial approximation factors. Chapter of [64].
71. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Proc. of STOC, pp. 84–93. ACM, New York (2005)
72. Regev, O.: The learning with errors problem, Invited survey in CCC 2010 (2010), http://www.cs.tau.ac.il/~odedr/
73. C. P. Schnorr. Progress on LLL and lattice reduction. Chapter of [64].
74. Schnorr, C.P.: A hierarchy of polynomial lattice basis reduction algorithms. Theor. Comput. Science 53, 201–224 (1987)
75. Schnorr, C.P., Euchner, M.: Lattice basis reduction: improved practical algorithms and solving subset sum problems. Mathematics of Programming 66, 181–199 (1994)

76. Schnorr, C.-P., Hörner, H.H.: Attacking the chor-rivest cryptosystem by improved lattice reduction. In: Guillou, L.C., Quisquater, J.-J. (eds.) EUROCRYPT 1995. LNCS, vol. 921, pp. 1–12. Springer, Heidelberg (1995)
77. Shoup, V.: NTL, Number Theory C++ Library, `http://www.shoup.net/ntl/`
78. Siegel, C.L.: Lectures on the Geometry of Numbers. Springer, Heidelberg (1989)
79. Sommer, N., Feder, M., Shalvi, O.: Finding the closest lattice point by iterative slicing. SIAM J. Discrete Math. 23(2), 715–731 (2009)
80. Stehlé, D., Watkins, M.: On the extremality of an 80-dimensional lattice. In: Hanrot, G., Morain, F., Thomé, E. (eds.) ANTS-IX. LNCS, vol. 6197, pp. 340–356. Springer, Heidelberg (2010)
81. Voronoi, G.: Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Journal für die reine und angewandte Mathematik 134, 198–287 (1908)
82. Voulgaris, P.: Personal communication
83. Wang, X., Liu, M., Tian, C., Bi, J.: Improved Nguyen-Vidick heuristic sieve algorithm for shortest vector problem. Cryptology ePrint Archive (2010), `http://eprint.iacr.org/2010/647`

A Appendix

The proofs for the optimal version of `AKS` and the improved version based on the birthday paradox have not been published in detail yet. For reference, we give a compact proof for all theorems of Section 4, largely based on [65,58,68].

A.1 Analysis of the AKS Algorithm

Lemma A.1 (Adapted from [58, Th. 4.1]). *Let $c_t = -\log_2 \gamma + 0.401$. If $\mathcal{T}$ is a set of points in $\mathcal{B}_n(R)$ such that the distance between two points is at least γR, then $|\mathcal{T}|$ is at most $N_T(n) = 2^{c_t n + o(n)}$. In particular, for any application of the* `Sieve` *function, the number of centers is at most $2^{c_t n + o(n)}$.*

Proof. Let $\alpha = 1 + \frac{1}{n}$. The ball $\mathcal{B}_n(\gamma R/3)$ contains at most one point. We cover $\mathcal{B}_n(R) \setminus \mathcal{B}_n(\gamma R/3)$ with coronas $T_r = \mathcal{B}_n(\alpha r) \setminus \mathcal{B}_n(r)$ for $r = \frac{\gamma R}{3}, \frac{\gamma R}{3}\alpha, \ldots, \frac{\gamma R}{3}\alpha^k$, with $k = \lceil n \log_2 \frac{3}{\gamma} \rceil = O(n)$. It suffices to prove that any corona T_r contains at most $2^{c_t n + o(n)}$ points.

Let $\boldsymbol{u}$ and $\boldsymbol{v}$ be two distinct points in $T_r \cap \mathcal{B}_n(R)$. We have $\langle \boldsymbol{u} - \boldsymbol{v}, \boldsymbol{u} - \boldsymbol{v} \rangle \geq (\gamma R)^2$, so $\langle \boldsymbol{u}, \boldsymbol{v} \rangle \leq \frac{1}{2}\left(\|\boldsymbol{u}\|^2 + \|\boldsymbol{v}\|^2 - (\gamma R)^2\right)$. We let $\phi_{\boldsymbol{u},\boldsymbol{v}}$ denote the angle between $\boldsymbol{u}$ and $\boldsymbol{v}$. The above implies that:

$$\begin{aligned}\cos \phi_{\boldsymbol{u},\boldsymbol{v}} = \frac{\langle \boldsymbol{u}, \boldsymbol{v} \rangle}{\|\boldsymbol{u}\| \cdot \|\boldsymbol{v}\|} &\leq \frac{1}{2}\left(\frac{\|\boldsymbol{u}\|}{\|\boldsymbol{v}\|} + \frac{\|\boldsymbol{v}\|}{\|\boldsymbol{u}\|} - \frac{(\gamma R)^2}{\|\boldsymbol{u}\| \cdot \|\boldsymbol{v}\|}\right) \\ &\leq 1 + \frac{1}{n} - \frac{(\gamma R)^2}{2R^2} = 1 + \frac{1}{n} - \frac{\gamma^2}{2} \underset{n \to \infty}{\longrightarrow} 1 - \frac{\gamma^2}{2}.\end{aligned}$$

For any $\varepsilon \in (0, \frac{\gamma^2}{2})$ and sufficiently large n we can apply Theorem 4.1 with $\phi_0 = \cos^{-1}\left(1 - \frac{\gamma^2}{2} + \varepsilon\right) \leq 60°$. This provides the result. □

Lemma A.2. *Let $c_b = \log_2 r_0 + 0.401$. For any lattice L, we have $|\mathcal{B}_n(r_0\lambda) \cap L| \leq N_B(n) = 2^{c_b n + o(n)}$.*

Proof. The distance between two lattice points is at least $\lambda(L)$. Let $\varepsilon > 0$ and $\gamma = \frac{1}{r_0(1+\varepsilon)}$. Provided that $n > \frac{1}{\varepsilon}$, we have $\frac{\lambda(L)}{r_0\lambda} \geq \gamma$. Applying Lemma A.1 with $R = r_0\lambda$ and γ for any $\varepsilon > 0$ provides the result. □

Lemma A.3 ([65, Th. 3.4]). *Let $c_g = -\frac{1}{2}\log_2(1 - \frac{1}{4\xi^2})$ and $\boldsymbol{s}$ be a shortest non-zero vector of L. Let $I_{\boldsymbol{s}} = \mathcal{B}_n(\mathbf{0}, \xi\lambda) \cap \mathcal{B}_n(-\boldsymbol{s}, \xi\lambda)$. If $\boldsymbol{x}$ is chosen uniformly in $\mathcal{B}_n(\mathbf{0}, \xi\lambda)$, then $\Pr(\boldsymbol{x} \in I_{\boldsymbol{s}}) \geq \frac{1}{N_G(n)}$ with $N_G(n) = 2^{c_g n + o(n)}$.*

Lemma A.4. *If* `AKS` *is run with $N \geq 1.01 N_G(n)(n^3 N_T(n) + N_B(n) + 1)$, then it returns a shortest non-zero vector with probability $\geq \frac{1}{2} - \underset{n\to\infty}{o}(1)$.*

Proof. From Lemma A.3, the initial set $\mathcal{T}$ contains at least $n^3 N_T(n) + N_B(n) + 1$ vectors such that $\boldsymbol{u}' - \boldsymbol{u} \in I_{\boldsymbol{s}} = \mathcal{B}_n(\mathbf{0}, \xi\lambda) \cap \mathcal{B}_n(-\boldsymbol{s}, \xi\lambda)$ with probability exponentially close to 1 as $n \to \infty$. Since no more than $\left\lceil \log_\gamma\left(\frac{\xi}{nR(1-\gamma)}\right)\right\rceil N_T(n) = O(n^2 N_T(n)) = o(n^3 N_T(n))$ vectors are used as centers and lost, this implies that the final set contains at least $N_B(n) + 1$ vectors such that $\boldsymbol{x} = \boldsymbol{u}' - \boldsymbol{u} \in I_{\boldsymbol{s}}$.

For $\boldsymbol{x} \in \mathcal{B}_n(\mathbf{0}w, \xi\lambda)$, let $\tau(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{s}$ if $\boldsymbol{x} \in I_{\boldsymbol{s}}$, $\tau(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{s}$ if $\boldsymbol{x} - \boldsymbol{s} \in I_{\boldsymbol{s}}$ and $\tau(\boldsymbol{x}) = \boldsymbol{x}$ otherwise. Consider `AKS2`, a modified version of `AKS` where τ is applied to each x_i with probability $\frac{1}{2}$ immediately after it is chosen. If $\boldsymbol{x}$ is sampled uniformly in $\mathcal{B}_n(\mathbf{0}, \xi\lambda)$, then so is $\tau(\boldsymbol{x})$. As a consequence, the outputs of `AKS` and `AKS2` follow the same distribution.

Since $\tau(\boldsymbol{x}) - \boldsymbol{x} \in L$, the lattice vector generated with the perturbation $\tau(\boldsymbol{x})$ is the same as the lattice vector generated with the perturbation $\boldsymbol{x}$. When the same random perturbations are chosen at the beginning, this implies that the behaviors of `AKS` and `AKS2` (test results and control flow) are exactly the same until the very last line of the algorithm. At the last line of `AKS`, at least one vector for which the perturbation is in $I_{\boldsymbol{s}}$ appears twice in $\mathcal{T}$ with probability exponentially close to 1. In `AKS2`, with probability close to $\frac{1}{2}$, the function τ is applied to exactly one of the two perturbations which have generated the lattice vector. If this occurs, the difference between the two lattice vectors corresponding to the same perturbation in `AKS2` is exactly $\boldsymbol{s}$. This implies that `AKS2` returns a correct result with probability arbitrarily close to $\frac{1}{2}$. Because of the property given above on the output distribution, this result also holds for `AKS`. □

Proof of Theorem 4.2. When the number of points is chosen according to Lemma A.4, the space complexity is $\leq 2^{c_{\text{space}} n + o(n)}$ with $c_{\text{space}} = c_g + \max(c_t, c_b)$. The time complexity for the sieve is $\leq 2^{(c_{\text{space}} + c_t)n + o(n)}$. The time complexity for the computation of pairwise differences is $\leq 2^{2c_b n + o(n)}$. So the global time complexity is $\leq 2^{c_{\text{time}} n + o(n)}$ with $c_{\text{time}} = \max(c_{\text{space}} + c_t, 2c_b)$. The optimal choice for constants ξ and γ to minimize the time complexity is $\gamma = 2 - \sqrt{2}$, $\xi = \frac{\sqrt{2}}{2}$ which leads to $c_{\text{space}} = 2.173$ and $c_{\text{time}} = 3.346$. Note that we have only proved that the algorithm succeeds with non-negligible probability, but running it n times ensures that it succeeds with probability exponentially close to 1 without significantly increasing the complexity. □

Improvement based on the birthday paradox. The number of sampled pairs if fixed to $N_G(n)(n^5 N_T(n)+n\sqrt{N_B(n)})$. Let $K = O(n^2)$ be the number of sieving steps. For each step, $\left\lfloor \frac{n^5}{K} N_G(n) N_T(n) \right\rfloor = \Omega\left(\frac{n^3}{K} N_G(n) N_T(n)\right)$ pairs are set aside to be used as centers.

At the beginning of the algorithm, among the pairs set aside to be used as centers for the first step, there are $\Omega(n^3 N_T(n))$ good pairs with high probability. As these pairs are processed, the probability that the distance between the next perturbed vector and the closest center is more than γR decreases. The sum of these probabilities is bounded from above by $N_T(n)$. As a consequence, once all centers have been processed, the probability for any of the subsequent pairs to be lost is $O\left(\frac{1}{n^3}\right)$. By induction, it can be proved the same proportion of pairs are lost at each step of the sieve with high probability. As a consequence, no more than $1-(1-\frac{1}{n^3})^{O(n^2)} = O\left(\frac{1}{n}\right)$ pairs are lost during the whole algorithm. This means that, in the final ball $\mathcal{B}_n(\mathbf{0}, \gamma R)$, there are $\Omega(n\sqrt{N_B(n)})$ (probabilistically) independent lattice points corresponding to good pairs with high probability. The birthday paradox implies that a collision occurs with probability close to 1. As in the proof of Lemma A.4, this implies that the algorithm returns a shortest vector with probability $\geq \frac{1}{2} - o(1)$.

A.2 Analysis of the ListSieve Algorithm

Lemma A.5 (Adapted from [58, Th. 4.1]). *Let $\xi > \frac{1}{2}$, $r_0 > 2\xi$ and $c_t = -\frac{1}{2}\log_2(1-\frac{2\xi}{r_0}) + 0.401$. At any moment during the execution of* `ListSieve`*, the list $\mathcal{T}$ contains at most $N_T(n) = 2^{c_t n + o(n)}$ vectors of norm at least $r_0\lambda$.*

Proof. We first bound the norm of any vector of $\mathcal{T}$. `NewPair` returns $(\boldsymbol{t}, \boldsymbol{t}')$ such that $\boldsymbol{t}' \in \mathcal{P}(B)$ and $\|\boldsymbol{t}' - \boldsymbol{t}\| \leq \xi\lambda$. We have assumed that $\max_i \|\boldsymbol{b}_i\| = 2^{O(n)}\lambda$. Hence $\|\boldsymbol{t}'\| \leq n \cdot \max_i \|\boldsymbol{b}_i\| \leq 2^{O(n)}\lambda$. After applying `Reduction`, the norm of $\boldsymbol{t}'$ does not increase and $\boldsymbol{t}' - \boldsymbol{t}$ is unchanged, so, for any $\boldsymbol{t}_i \in \mathcal{T}$, we have $r_0\lambda \leq \|\boldsymbol{t}_i\| \leq (2^{O(n)} + \xi)\lambda$. It now suffices to prove that any $\mathcal{T}_r = \{\boldsymbol{t}_i \in \mathcal{T} \mid r\lambda \leq \|\boldsymbol{t}_i\| \leq \left(1+\frac{1}{n}\right) r\lambda\}$ for $r \geq r_0$ contains at most $2^{c_t n + o(n)}$ points. Indeed, the list $\mathcal{T}$ is contained in a union of $O(n^2)$ sets $\mathcal{T}_r$.

Let $i < j$ such that $\boldsymbol{t}_i, \boldsymbol{t}_j \in \mathcal{T}_r$. The idea of the proof is that for large n, the angle between $\boldsymbol{t}'_j$ and $\boldsymbol{t}_i$ is not far from being above $\frac{\pi}{3}$ because $\boldsymbol{t}_i$ was already in $\mathcal{T}$ when $\boldsymbol{t}_j$ was reduced. We use the inequality $\|\boldsymbol{t}_j - \boldsymbol{t}'_j\| \leq \xi\lambda$ to obtain a lower bound for the angle $\phi_{\boldsymbol{t}_i, \boldsymbol{t}_j}$ between $\boldsymbol{t}_i$ and $\boldsymbol{t}_j$ and then apply Theorem 4.1.

Note that $\|\boldsymbol{t}'_j\| \leq \|\boldsymbol{t}_j\| + \xi\lambda \leq 3r\lambda$. Since $\boldsymbol{t}_j$ was added after $\boldsymbol{t}_i$, we have:

$$\|\boldsymbol{t}'_j - \boldsymbol{t}_i\|^2 > \left(1-\frac{1}{n}\right)^2 \|\boldsymbol{t}'_j\|^2 \geq \left(1-\frac{2}{n}\right)\|\boldsymbol{t}'_j\|^2$$

$$\langle \boldsymbol{t}'_j, \boldsymbol{t}_i \rangle < \frac{1}{2}\left[\|\boldsymbol{t}_i\|^2 + \frac{2}{n}\|\boldsymbol{t}'_j\|^2\right] \leq \frac{1}{2}\|\boldsymbol{t}_i\|^2 + \frac{1}{n}(3r\lambda)^2.$$

Moreover, we have $\langle \boldsymbol{t}_j - \boldsymbol{t}'_j, \boldsymbol{t}_i \rangle \leq \xi\lambda\|\boldsymbol{t}_i\|$. We can now bound $\cos(\phi_{\boldsymbol{t}_i,\boldsymbol{t}_j})$.

$$\begin{aligned}
\langle \boldsymbol{t}_j, \boldsymbol{t}_i \rangle = \langle \boldsymbol{t}'_j, \boldsymbol{t}_i \rangle + \langle \boldsymbol{t}_j - \boldsymbol{t}'_j, \boldsymbol{t}_i \rangle &\leq \frac{1}{2}\|\boldsymbol{t}_i\|^2 + \frac{1}{n}(3r\lambda)^2 + \xi\lambda\|\boldsymbol{t}_i\| \\
\cos(\phi_{\boldsymbol{t}_i,\boldsymbol{t}_j}) = \frac{\langle \boldsymbol{t}_j, \boldsymbol{t}_i \rangle}{\|\boldsymbol{t}_i\| \cdot \|\boldsymbol{t}_j\|} &\leq \frac{1}{2}\frac{\|\boldsymbol{t}_i\|}{\|\boldsymbol{t}_j\|} + \frac{1}{n} \cdot \frac{(3r\lambda)^2}{\|\boldsymbol{t}_i\| \cdot \|\boldsymbol{t}_j\|} + \frac{\xi\lambda}{\|\boldsymbol{t}_j\|} \\
&\leq \frac{1}{2}\left(1 + \frac{1}{n}\right) + \frac{9}{n} + \frac{\xi}{r} \\
&\leq \frac{1}{2} + \frac{\xi}{r_0} + O\left(\frac{1}{n}\right).
\end{aligned}$$

The bound on $|\mathcal{T}_r|$ follows directly from Theorem 4.1. □

Lemma A.6. *Let $c_t = \log_2(\xi + \sqrt{\xi^2+1}) + 0.401$. At any moment during the execution of* `ListSieve`*, the list $\mathcal{T}$ contains at most $N_T(n) = 2^{c_t n + o(n)}$ vectors.*

Proof. Let $r_0 > 2\xi$. Since the list $\mathcal{T}$ contains only lattice points, one can combine Lemma A.5 with Lemma A.2 to obtain the upper bound $|\mathcal{T}| \leq 2^{\max(c_t,c_b)n+o(n)}$. This upper bound is minimized when $c_t = c_b$, which occurs when $r_0 = \xi + \sqrt{\xi^2+1}$. □

Proof of Theorem 4.3 [Adapted from [58, Sec. 4.1]]. Let $I_{\boldsymbol{s}} = \mathcal{B}_n(\boldsymbol{0}, \xi\lambda) \cap \mathcal{B}_n(-\boldsymbol{s}, \xi\lambda)$, $I_{-\boldsymbol{s}} = I_{\boldsymbol{s}} + \boldsymbol{s}$ and $J = I_{\boldsymbol{s}} \cup I_{-\boldsymbol{s}}$. A pair $(\boldsymbol{u}, \boldsymbol{u}')$ is *good* if the perturbation $\boldsymbol{x} = \boldsymbol{u}' - \boldsymbol{u}$ is in J. The number of points in $\mathcal{T}$ is bounded by $N_T(n)$. All perturbations are sampled independently with uniform distribution and the probability that a perturbation $\boldsymbol{x}$ is in J is at least $2N_G(n)^{-1}$ (Lemma A.3). For $\boldsymbol{x} \in \mathcal{B}_n(0, \xi\lambda)$, we define $\tau(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{s}$ if $\boldsymbol{x} \in I_{\boldsymbol{s}}$, $\tau(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{s}$ if $\boldsymbol{x} \in I_{-\boldsymbol{s}}$ and $\tau(\boldsymbol{x}) = \boldsymbol{x}$ otherwise (τ is well defined provided that $\xi < 1$). We define the following events:

- E_i: "the lattice point produced by the reduction of the i-th good pair is added to the list" (assuming that we stop the algorithm when at least i good pairs have been sampled);
- F: "`ListSieve` returns a shortest non-zero vector";
- G: "at least $2N_T(n)$ good pairs are sampled".

When $N = 1.01N_T(n)N_G(n)$, $\Pr(G)$ tends to 1 as n increases, so if n is large enough $\Pr(G) \geq \frac{1}{2}$. The fact that $\sum_{i=1}^{2N_T(n)} \Pr(E_i|G) \leq N_T(n)$ implies that for some $i \leq 2N_T(n)$, we have $\Pr(E_i|G) \leq \frac{1}{2}$.

If $\boldsymbol{x}_i$ is sampled with uniform distribution in J, so is $\tau(\boldsymbol{x}_i)$. Consider the i-th good pair. Assume that $\boldsymbol{x}_i$ is such that after the reduction of the pair $(\boldsymbol{u}_i, \boldsymbol{u}'_i)$, the vector $\boldsymbol{u}_i$ is already in the list. With the perturbation $\tau(\boldsymbol{x}_i) = \boldsymbol{x}_i \pm \boldsymbol{s}$, the pair would be $(\boldsymbol{u}_i \pm \boldsymbol{s}, \boldsymbol{u}'_i)$ so the reduction of the pair would produce $\boldsymbol{t} \pm \boldsymbol{s}$. It might or might not already be in the list. In both cases, the algorithm succeeds. This implies that $\Pr(F|G) \geq \Pr\left(\bar{E}|G\right) \geq \frac{1}{2}$ so $\Pr(F) \geq \Pr(F|G)\Pr(G) \geq \frac{1}{4}$. □

A.3 Analysis of the ListSieveBirthday Algorithm

What follows is a short version of [68]. The bound $N_T(n)$ from Lemma A.5 holds for `ListSieveBirthday` and is an upper bound on $|\mathcal{T}|$ The first part of the proof of Theorem 4.4 consists in proving that $|\mathcal{U}|$ is large enough with non-negligible probability. It relies on Lemmas A.2 and A.3, where $N_B(n)$, $N_G(n)$ and $I_{\boldsymbol{s}}$ are defined.

Lemma A.7. *Let* $N_1^{\max} = 4N_G(n)N_T(n)$. *Consider* `ListSieveBirthday` *with* $\xi > \frac{1}{2}$, $r_0 > 2\xi$ *and* $N_1 = N_1^{\max}$. *For* $i \leq N_1^{\max}$, *we define the event* $E_i : \|\boldsymbol{t}_i\| < r_0\lambda$. *We let* $p_i = \Pr(E_i \mid \boldsymbol{x}_i \in I_{\boldsymbol{s}})$, *where the probability is taken over the randomness of* $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i$, *and* $J = \{i \leq N_1^{\max} : p_i \leq \frac{1}{2}\}$. *Then* $|J| \leq N_1^{\max}/2$.

Proof. Assume (for contradiction) that $|J| > N_1^{\max}/2$. Then Lemma A.3 implies that $\sum_{i\in J}(1-p_i)\Pr(\boldsymbol{x}_i \in I_{\boldsymbol{s}}) \geq \frac{|J|}{2N_G} > N_T$. This contradicts the following (the last inequality comes from the bound $|\mathcal{T}| \leq N_T(n)$):

$$\begin{aligned}\sum_{i\in J}(1-p_i)\Pr(\boldsymbol{x}_i \in I_{\boldsymbol{s}}) &= \sum_{i\in J}\Pr(\neg E_i \cap (\boldsymbol{x}_i \in I_{\boldsymbol{s}})) \\ &\leq \sum_{i\geq 1}\Pr(\neg E_i) \leq N_T. \qquad \square\end{aligned}$$

In the second loop of `ListSieveBirthday`, we do not add any point to $\mathcal{T}$. Therefore, the points that are added to $\mathcal{U}$ are iid. The procedure to reduce points being the same in both loops, we have that for any $i \leq N_2$ such that $\boldsymbol{y}_i \in I_{\boldsymbol{s}}$, the probability that $\|\boldsymbol{u}_i\| < r_0\lambda$ is p_{N_1+1}. When N_1 is sampled uniformly in $[0, N_1^{\max}-1]$, we have $p_{N_1+1} \geq \frac{1}{2}$ with probability $\geq \frac{1}{2}$, by Lemma A.7.

Lemma A.8. *If* n *is sufficiently large, then with probability* $\geq 1/4$ *(taken over the randomness of* N_1, *the* x_k*'s and the* y_k*'s), there exist two distinct indices* $i, j \leq N_2$ *such that* $\boldsymbol{u}_i = \boldsymbol{u}_j$ *and* $\boldsymbol{y}_i, \boldsymbol{y}_j \in I_{\boldsymbol{s}}$.

Proof. In this proof, we assume that $p_{N_1+1} \geq \frac{1}{2}$. This occurs with probability $\geq \frac{1}{2}$ and implies that $\Pr(\|\boldsymbol{u}_i\| \leq r_0\lambda \mid \boldsymbol{y}_i \in I_{\boldsymbol{s}}) \geq \frac{1}{2}$ for all $i \leq N_2$. Let $X = |\{i \leq N_2 : (\|\boldsymbol{u}_i\| \leq r_0\lambda) \cap (\boldsymbol{y}_i \in I_{\boldsymbol{s}})\}|$. Lemma A.3 gives

$$\begin{aligned}\Pr((\|\boldsymbol{u}_i\| \leq r_0\lambda) \cap (\boldsymbol{y}_i \in I_{\boldsymbol{s}})) &= \Pr(\|\boldsymbol{u}_i\| \leq r_0\lambda \mid \boldsymbol{y}_i \in I_{\boldsymbol{s}})\Pr(\boldsymbol{y}_i \in I_{\boldsymbol{s}}) \\ &\geq \frac{1}{2N_G}.\end{aligned}$$

Let $N = 2\lceil\sqrt{N_B}\rceil$. The variable X has a binomial distribution of parameter $p \geq \frac{1}{2N_G}$. We have $\mathbb{E}(X) = pN_2 \geq 2N$ and $\mathrm{Var}(X) = p(1-p)N_2 \leq \mathbb{E}(X)$. Therefore, by using Chebyshev's inequality, we have (since $N_B \geq 25$ holds for n large enough, we have $N \geq 10$):

$$\begin{aligned}\Pr(X \leq N) &\leq \Pr(|X - \mathbb{E}(X)| \geq \mathbb{E}(X) - N) \\ &\leq \frac{\mathrm{Var}(X)}{(\mathbb{E}(X)-N)^2} \leq \frac{\mathbb{E}(X)}{(\mathbb{E}(X)-N)^2} \leq \frac{2}{N} \leq \frac{1}{5}.\end{aligned}$$

So with probability $\geq \frac{4}{5}$, `ListSieveBirthday` samples at least N iid points in $S_0 = \mathcal{B}_n(r_0\lambda) \cap L$. The probability that a collision occurs is minimized when the distribution is uniform, i.e., the probability of each point is $1/|S_0|$. Since we have chosen $N \geq \sqrt{|S_0|}$ (by Lemma A.2), the birthday paradox implies that the probability that i and j exist will be large. More precisely it is bounded from below by

$$\frac{4}{5}\left(1 - \prod_{i<N}\left(1 - \frac{i}{|S_0|}\right)\right) \geq \frac{4}{5}\left(1 - \mathrm{e}^{-\frac{N(N-1)}{2N_B}}\right) \geq \frac{4}{5}\left(1 - \frac{1}{\mathrm{e}}\right) \geq \frac{1}{2},$$

where we used the fact that $|S_0| \leq N_B(n)$ (by Lemma A.2). □

Proof of Theorem 4.4. Introducing a modification `ListSieveBirthday2` of `ListSieveBirthday` that applies τ to every y_i with probability $\frac{1}{2}$ and using the same reasoning as in the proof of Lemma A.4 leads to the fact that the algorithm succeeds with probability $\geq \frac{1}{8}$.

The space complexity is $|\mathcal{T}| + |\mathcal{U}|$. We have $|\mathcal{T}| \leq 2^{c_t n + o(n)}$, and, by definition of N_2, we have $|\mathcal{U}| \leq 2^{(c_g + c_b/2)n + o(n)}$. Since $\|\boldsymbol{b}_i\| = 2^{O(n)}\lambda$ for all i, the complexity of `Reduction` is $|\mathcal{T}|\mathcal{P}oly(n, |B|)$. Omitting the polynomial factor, the time complexity of the first loop is $|\mathcal{T}|N_1 \leq |\mathcal{T}|N_1^{\max} \leq 2^{(c_g + 2c_t)n + o(n)}$. The time required to find a closest pair of points in $\mathcal{U}$ with the naive algorithm is $|\mathcal{U}|^2$. Finally, the time complexity of the second loop is $|\mathcal{T}| \cdot |\mathcal{U}| \leq 2^{(c_t + c_g + c_b/2)n + o(n)}$, which is negligibly smaller than the cost of one of the other components.

The time complexity bound is minimized when $2c_t = c_g + c_b$. By Lemmas A.2, A.5 and A.3, this is equivalent to $r_0 = 2\xi + 2^{0.401}\sqrt{1 - \frac{1}{4\xi^2}}$. Optimizing with respect to ξ leads to $\xi \simeq 0.9476$, $r_0 \simeq 3.0169$, $c_{\text{time}} \leq 2.465$ and $c_{\text{space}} \leq 1.233$. Calling the algorithm n times allows to ensure that it succeeds with probability exponentially close to 1. □

An Experiment of Number Field Sieve over GF(p) of Low Hamming Weight Characteristic

Kenichiro Hayasaka[1] and Tsuyoshi Takagi[2]

[1] Graduate School of Mathematics, Kyushu University,
[2] Institute of Mathematics for Industry, Kyushu University,
744, Motooka, Nishi-ku, Fukuoka, 819-0395, Japan

Abstract. The security of the digital signature algorithm (DSA) and Diffie-Hellman key exchange is based on the difficulty of the discrete logarithm problems (DLP) over prime field GF(p), and thus it is important to evaluate the difficulty of the DLP over GF(p) for discussing the security of these protocols. The number field sieve (NFS) is asymptotically the fastest algorithm to solve the DLP over GF(p). NFS was first proposed by Gordon and then it was improved by Schirokauer and Joux-Lercier. On the other hand, Schirokauer presented a new variant of NFS, which is particularly efficient for the characteristic p with low weight (p has a signed binary representation of low Hamming weight). In this paper, we implement the NFS proposed by Joux-Lercier and Schirokauer, and then we compare the running time of the NFS using the polynomials by Joux-Lercier and Schirokauer with respect to low weight primes of 100 bits or 110 bits.

Keywords: discrete logarithm, number field sieve, low Hamming weight.

1 Introduction

The security of the digital signature algorithm (DSA) or the Diffie-Hellman key exchanging scheme used in PGP is based on the hardness of the discrete logarithm problem (DLP) over prime field GF(p) for a large prime p. The number field sieve (NFS) is the asymptotically fastest algorithm for solving the discrete logarithm problem over GF(p). The expected running time of the NFS is asymptotically

$$L_p[1/3; c] = \exp((c + o(1))(\log p)^{1/3}(\log\log p)^{2/3}), \tag{1}$$

where c is a constant and $o(1) \to 0$ for $p \to \infty$.

The number field sieve was initially developed for factoring integers [10]. Gordon first applied the NFS to solve the DLP over GF(p) [3], and Schirokauer then achieved the running time as fast as the general number field sieve, i.e., $L_p[1/3; (64/9)^{1/3}]$ using the character map [16]. Joux and Lercier introduced the virtual logarithms for the NFS [4,17]. There are several experiments on solving the DLP over GF(p) [12], and the current largest size of characteristic is

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 191–200, 2011.
© Springer-Verlag Berlin Heidelberg 2011

of 530 bits by Kleinjung *et al.* [6]. The running time of the NFS depends on the definition polynomial of the underlying number field [5,13], and the experiment by Kleinjung *et al.* [6] used the skewed polynomial of degree 5 generated by the based-m method. Moreover, Joux-Lercier extended the Gaussian integer method [2,7] to the pair of polynomials of degree d and $d+1$ [4], which is called JL03-NFS in this paper.

On the other hand, if the Hamming weight of p is low (p is represented by the summation of a few $\pm 2^e$), the running time of modular arithmetic modulo p can be accelerated [20]. We are able to improve the processing speed of cryptosystem constructed over GF(p) with such a prime p of low weight. Schirokauer proposed a new class of definition polynomials for NFS over GF(p) of low-weight p that makes the constant term c of equation (1) smaller than $(64/9)^{1/3}$ in [18,19]. This version of NFS is called Sch06-NFS.

To the best of our knowledge there is no report about practical experiment on the running time of Sch06-NFS. In this paper, we implement both JL03-NFS and Sch06-NFS on a PC, and show a comparison of the running time between JL03-NFS and Sch06-NFS for the same prime p of low Hamming weight.

This paper is organized as follows. Section 2 gives an overview of the number field sieve. Section 3.1 explains the polynomial selection step proposed by Joux and Lercier [4]. Section 3.2 describes a polynomial selection step proposed by Schirokauer [18,19]. We then present an experiment on both JL03-NFS and Sch06-NFS on a PC and give their comparison in Section 4. Finally we state some concluding remarks in Section 5.

2 Number Field Sieve

In this section, we give an overview of the number field sieve that consists of four steps: polynomial selection, searching relations, linear algebra, and solving the individual logarithm.

2.1 DLP over GF(p)

Denote by GF(p) be a finite field of p elements. Let g be a generator of the multiplicative group GF(p)$^\times$.

The discrete logarithm problem (DLP) is a problem that for a target element $a \in \mathrm{GF}(p)^\times$ we try to find $x \in \{1, 2, \ldots, p-1\}$ such that $g^x = a$. The x is called the discrete logarithm of a to the base g and we write x as $\log_g a$. In this paper we assume that p satisfies $p - 1 = 2\ell$ for some prime ℓ.

2.2 Polynomial Selection

Let $f_\beta(X) = \sum_{i=0}^{d_\beta} c_{\beta,i} X^i$ and $f_\alpha(X) = \sum_{i=0}^{d_\alpha} c_{\alpha,i} X^i$ be polynomials in $\mathbb{Z}[X]$ of degree d_β and d_α, respectively. Here we assume that f_β is monic ($c_{\beta,d_\beta} = 1$) and f_α is non-monic ($c_{\alpha,d_\alpha} \neq 1$). We then assume that polynomials f_β and f_α are irreducible over $\mathbb{Q}$ and satisfy $f_\beta(m) \equiv f_\alpha(m) \equiv 0 \bmod p$ for prime p and

$m \in \mathrm{GF}(p)^\times$. Denote by β and α the root of $f_\beta(X) = 0$ and $f_\alpha(X) = 0$ in $\mathbb{C}$, respectively. Let $\mathcal{O}_\beta$ be the ring of integers of $\mathbb{Q}(\beta)$. We deal with the order $\mathbb{Z}[\beta] \subset \mathcal{O}_\beta$ and a ring homomorphism

$$\phi_\beta : \mathbb{Z}[\beta] \to \mathrm{GF}(p), \quad \beta \mapsto m \bmod p. \tag{2}$$

Let $\mathcal{O}_\omega$ be the ring of integers of $\mathbb{Q}(\omega)$ for the root $\omega = c_{\alpha,d_\alpha}\alpha$ of monic polynomial $f_\omega(X) = c_{\alpha,d_\alpha}^{d_\alpha-1} f_\alpha(X/c_{\alpha,d_\alpha})$ we also deal with the order $\mathbb{Z}[\omega] \subset \mathcal{O}_\omega$ and a ring homomorphism

$$\phi_\omega : \mathbb{Z}[\omega] \to \mathrm{GF}(p), \quad \omega \mapsto c_{\alpha,d_\alpha} m \bmod p. \tag{3}$$

2.3 Searching Relations

For real number $B > 0$, we define the set of prime ideals of degree one as follows.

$$B_\beta = \{(q, \beta - t) \mid q \in \mathbb{P}, q \le B, f_\beta(t) \equiv 0 \bmod q\},$$
$$B_\omega = \{(q, \omega - t) \mid q \in \mathbb{P}, q \le B, f_\omega(t) \equiv 0 \bmod q\}.$$

where $\mathbb{P} = \{2, 3, 5, 7, \ldots\}$ is the set of all primes. The upper-bound B appeared in B_β, B_ω is called the smoothness bound. If the prime factors of an integer z are less than or equal to B, then z is called B-smooth.

For integers a, b the norm of ideal $a + b\alpha$ or $a + b\beta$ is defined by $N(a + b\alpha) = (-b)^{d_\alpha} f_\alpha(-a/b)$ or $N(a + b\beta) = (-b)^{d_\beta} f_\beta(-a/b)$, respectively. Let S be the set of all pairs (a, b) in $|a| \le C, 0 < b \le C$ for a real number $C > 0$ that satisfy the following three conditions:

(i) $\gcd(a, b) = 1$,
(ii) $|N(a + b\beta)| = |(-b)^{d_\beta} f_\beta(-a/b)|$ is B-smooth,
(iii) $|N(a + b\alpha)| = |(-b)^{d_\alpha} f_\alpha(-a/b)|$ is B-smooth.

We try to find many pairs $(a, b) \in S$ such that $\#S > \#B_\beta + \#B_\omega + r_\beta + r_\omega$, where r_β and r_ω are the torsion-free rank of $\mathcal{O}_\beta^\times$ and $\mathcal{O}_\omega^\times$ used for Schirokauer's character map [16]. The real number C is called the sieving interval.

Next we explain how to find a relation corresponding to one element $(a, b) \in S$ in the following. For prime q with $q \nmid [\mathcal{O}_\beta : \mathbb{Z}[\beta]]$, the exponent ε of prime q appeared in the prime decomposition of $N(a + b\beta)$ is exactly equal to that of the prime ideal $(q, \beta - t) \in B_\beta$ in the prime ideal decomposition of $(a + b\beta)$, namely

$$q^{\varepsilon_\beta} \mid\mid N(a + b\beta) \quad \Rightarrow \quad (q, \beta - t)^{\varepsilon_\beta} \mid\mid (a + b\beta), \tag{4}$$

where $t \equiv -a/b \bmod q$. The treatment of the bad prime q with $q \mid [\mathcal{O}_\beta : \mathbb{Z}[\beta]]$ can be found in Section 15.5 of [1]. Therefore we can obtain $\varepsilon_{(q,\beta-t)}$ such that

$$(a + b\beta)\mathcal{O}_\beta = \prod_{(q,\beta-t)\in B_\beta} (q, \beta - t)^{\varepsilon_{(q,\beta-t)}}. \tag{5}$$

On the other hand, the norm of the non-monic polynomial f_α is different from the monic case, namely the norm of any ideal $(c_{\alpha,d_\alpha}a + b\omega)$ for $a, b \in \mathbb{Z}$ satisfies $N(c_{\alpha,d_\alpha}a + b\omega) = c_{\alpha,d_\alpha}^{d_\alpha - 1} N(a + b\alpha)$. Let $\mathfrak{c}$ be an ideal in $\mathcal{O}_\omega$ whose norm is equal to $c_{\alpha,d_\alpha}^{d_\alpha-1}$. Then from condition (iii) we have the following decomposition

$$(c_{\alpha,d_\alpha}a + b\omega)\mathcal{O}_\omega = \mathfrak{c} \prod_{(q,\omega-t)\in B_\omega} (q, \omega - t)^{\varepsilon_{(q,\omega-t)}}. \tag{6}$$

Similar to the case of prime ideals in B_β we can determine the exponent ε_ω of prime q of $N(a + b\alpha)$ by

$$q^{\varepsilon_\omega} \mid\mid N(a + b\alpha) \quad \Rightarrow \quad (q, \omega - t)^{\varepsilon_\omega} \mid\mid (c_{\alpha,d_\alpha}a + b\omega)\mathfrak{c}^{-1},$$

where $(q, \omega - t) \in B_\omega$ and $t \equiv -c_{\alpha,d_\alpha}a/b \bmod q$.

Here note that $\phi_\beta(c_{\alpha,d_\alpha}(a + b\beta)\mathcal{O}_\beta) \equiv \phi_\omega((c_{\alpha,d_\alpha}a + b\omega)\mathcal{O}_\omega) \bmod p$ holds due to maps ϕ_β and ϕ_ω in Section 2.2. In order to have an equation of logarithms as element (not ideal), we add the Schirokauer's character map $(\lambda_{\beta,1}, \lambda_{\beta,2}, ..., \lambda_{\beta,r_\beta})$ and $(\lambda_{\omega,1}, \lambda_{\omega,2}, ..., \lambda_{\omega,r_\omega})$, which can be computed by f_β, f_ω, (a, b) and ℓ [16]. From equations (5) and (6) we obtain the following relation

$$\begin{aligned} y + \sum_{(q,\beta-t)\in B_\beta} \varepsilon_{(q,\beta-t)} \log \phi_\beta((q, \beta - t)) + \sum_{i=1}^{r_\beta} \lambda_{\beta,i} x_{\beta,i} \equiv \\ \sum_{(q,\omega-t)\in B_\omega} \varepsilon_{(q,\omega-t)} \log \phi_\omega((q, \omega - t)) + \sum_{j=1}^{r_\omega} \lambda_{\omega,j} x_{\omega,j} \mod \ell, \end{aligned} \tag{7}$$

where $\log \phi_\beta((q, \beta - t)), \log \phi_\omega((q, \omega - t)), x_{\beta,i}, x_{\omega,j}$ are unknown variables, the modulus is $\ell = (p-1)/2$, and $y = \log(\phi_\beta(c_{\alpha,d_\alpha})/\phi_\omega(\mathfrak{c}))$ is independent from the choice of $(a, b) \in S$.

2.4 Linear Algebra and Individual Logarithm

We explain how to solve the discrete logarithms $\log_g q$ for the base g and prime q smaller than the smoothness bound B.

Here we assume that both prime q and base g are completely split in $\mathcal{O}_\beta$ or $\mathcal{O}_\omega$ and we set $\zeta = \beta$ or ω, respectively. The base g of the discrete logarithm in equation (7) can be fixed to g such as $\log_g$ by adding the following equation

$$\sum_{\mathfrak{g}\in B_\zeta, \mathfrak{g}\mid g} \log_g \mathfrak{g} + \sum_{i=1}^{r_\zeta} \lambda_{\zeta,i} x_{\zeta,i} \equiv 1 \bmod \ell.$$

After solving the linear equations (7) with the fixed base g we obtain the values $\log_g \phi_\beta((q, \beta - t)), \log_g \phi_\omega((q, \omega - t))$ for $(q, \beta - t) \in B_\beta, (q, \omega - t) \in B_\omega$ and $x_{\beta,i}, x_{\omega,j}$ for $i = 1, 2, ..., r_\beta$, $j = 1, 2, ..., r_\omega$.

Finally $\log_g q \bmod \ell$ can be computed by

$$\log_g q \equiv \sum_{\substack{(q,\zeta-t)\in B_\zeta,\\ (q,\zeta-t)\ |q}} \varepsilon_{(q,\zeta-t)} \log_g \phi_\zeta((q,\zeta-t)) + \sum_{i=1}^{r_\zeta} \lambda_{\zeta,i} x_{\zeta,i} \bmod \ell. \tag{8}$$

The individual discrete logarithm $\log_g a$ for any target $a \in \mathrm{GF}(p)^\times$ can be efficiently computed by the special-$\mathfrak{q}$ descent technique [4].

3 Joux-Lercier's NFS and Low Weight NFS

In this section we explain two efficient variations of NFS proposed by Joux and Lercier [4] and Schirokauer [18].

3.1 Joux-Lercier's NFS (JL03-NFS)

Joux and Lercier proposed an efficient polynomial selection step [4], and we denote the NFS using this definition polynomial by JL03-NFS.

JL03-NFS generates two polynomials f_β of degree $d+1$ and f_α of degree d in $\mathbb{Z}[X]$ for some $d \in \mathbb{N}$, where f_β and f_α have a common root $\bmod p$. Here f_β is a monic polynomial whose coefficients are small. The polynomial f_α is non-monic and the size of all $d+1$ coefficients of f_α is as large as $p^{1/(d+1)}$.

We can generate such two polynomials as follows. Generate a polynomial f_β in $\mathbb{Z}[X]$ of degree $d+1$, which is irreducible over $\mathbb{Q}$ but has a root m such that $f_\beta(m) = 0 \bmod p$. For example, we test whether the polynomial $f_\beta(X) = X^{d+1} + a$ satisfies the above conditions for $a = 2, 3, 4, \ldots$.

Next, we explain how to generate a polynomial f_α of degree d whose root $\bmod p$ is m. Let A be a $(d+1) \times (d+1)$ matrix

$$A = \begin{pmatrix} p & -m \bmod p & -m^2 \bmod p & \cdots & -m^d \bmod p \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}.$$

After applying a lattice reduction algorithm [11] to A, we set the polynomial $f_\alpha(X) = \sum_{i=0}^{d} m_i X^i$ of degree d for the first column $(m_0, m_1, \ldots, m_d)^T$ of the reduced matrix. This polynomial satisfies that $f_\alpha(m) \equiv 0 \bmod p$ and $|m_i| \approx p^{1/(d+1)}$ $(i = 0, 1, \ldots, d)$ due to the property of the reduced basis.

3.2 Low Weight NFS (Sch06-NFS)

Schirokauer proposed a polynomial selection step for the number field sieve over $\mathrm{GF}(p)$ of low-weight prime p [18], and we denote the NFS using this polynomial by Sch06-NFS.

For odd prime p the weight of p is the small integer w such that

$$p = \sum_{i=1}^{w} \epsilon_i 2^{c_i}, \tag{9}$$

where $\epsilon_1, \epsilon_2, \ldots, \epsilon_w \in \{-1, 1\}$ and $c_1, c_2, \ldots, c_w$ are integers that satisfy the condition $c_w > c_{w-1} > \ldots > c_1 = 0$.

We explain how to generate a polynomial f_α of degree d which has a common root mod$p$ with $f_\beta(X) = X - m$ but the coefficients of f_α are still small. Let $\theta = (2w-3)/(w-1)$ for the weight w of prime p. Let $e = \lfloor c_w/d \rfloor$ and

$$\bar{c}_i = c_i \bmod e \quad \text{for } i = 1, 2, \ldots, w, \tag{10}$$

and $\bar{c}_{w+1} = e$. Then σ is a permutation map over $\{1, 2, \ldots, w+1\}$ that satisfies

$$0 = \bar{c}_{\sigma(1)} \leq \bar{c}_{\sigma(2)} \leq \cdots \leq \bar{c}_{\sigma(w)} \leq \bar{c}_{\sigma(w+1)} = e. \tag{11}$$

Here denote by J the index i which has the maximal difference $\bar{c}_{\sigma(i+1)} - \bar{c}_{\sigma(i)}$ of consecutive terms in inequality (11). If there exist more than one index of the maximal distance, then the larger i is chosen for J. We set $\Delta = \bar{c}_{\sigma(J+1)} - \bar{c}_{\sigma(J)}$ and $\mu = e - \bar{c}_{\sigma(J+1)}$. We define $f_\alpha(X) = \sum_{i=1}^{w} \epsilon_i 2^{a_i} x^{b_i}$ where a_i and b_i satisfy

$$\begin{cases} a_i = \bar{c}_i + \mu, \ b_i = \lfloor c_i/e \rfloor, & \text{if } \sigma^{-1}(i) \leq J, \\ a_i = \bar{c}_i - \bar{c}_{\sigma(J+1)}, \ b_i = \lfloor c_i/e \rfloor + 1, & \text{otherwise.} \end{cases}$$

Then $f_\alpha(X)$ is an irreducible polynomial over $\mathbb{Q}$ whose root mod$p$ is $m = 2^e$, namely f_α has a common root with $f_\beta(X) = X - m$.

3.3 Running Time

If the weight of prime p is small, the norm $N(a + b\alpha)$ using f_α remains relatively small for $(a, b) \in S$ and thus Sch06-NFS is more efficient than JL03-NFS. Indeed the maximal value of the coefficient a_i of f_α is $\max|a_i| = \hat{c}_{\sigma(J)} + \mu = e - \Delta$, which becomes small if the maximal distance Δ is close to e. This happens if all the distances of two consecutive non-zero bits of prime p in equation (9) become closer to the multipliers of e.

Schirokauer [18] estimated the running time of Sch06-NFS is asymptotically $L_p[1/3; (32\theta/9)^{1/3}]$ for $\theta = (2w-3)/(w-1)$ and the following parameters (degree d, smoothness bound B, sieve interval C) are optimal

$$d = \left(\frac{(3\theta + o(1)) \log p}{2 \log \log p} \right)^{1/3}, \tag{12}$$

$$B = C = L_p[1/3; (4\theta/9)^{1/3}]. \tag{13}$$

In the interval $1 \leq \theta < 2$, the running time of Sch06-NFS is at least as fast as that of JL03-NFS, namely $L_p[1/3; (64/9)^{1/3}]$ for $\theta = 2$. On the other hand, in the smallest case of $w = 2$, the running time of Sch06-NFS becomes $L_p[1/3; (32/9)^{1/3}]$, which is asymptotically as fast as that of the special number field sieve.

4 An Experiment on JL03-NFS and Sch06-NFS

In this section we compare JL03-NFS and Sch06-NFS for low-weight primes by implementing them on a PC.

4.1 Experimental Environments

Our experiment is carried on a Core2 Duo E8400 (3.00GHz) with 2GB RAM. We use gcc 4.3.3 with option `-O2` on Ubuntu Linux 8.10 (32 bits), and gnu mp 4.3.1 and NTL 5.5.2 are deployed for multi-precision integers. Due to the computational resources of our experimental environment we choose primes of either 100 bits or 110 bits. Moreover we choose primes p of weight less than or equal to 5 that satisfy $(p-1)/2$ is also prime. The relations are gathered by the line sieve and the decomposition of prime ideals over bad primes are computed by PARI [14]. Lanczos method is used for the linear algebra [9], but we do not implement the structured Gaussian elimination methods [8,15].

4.2 Parameters for JL03-NFS and Sch06-NFS

JL03-NFS requires two polynomials of degree d and $d+1$. From our initial experiment it was fastest to choose $d=1$ for primes of about 100 bits. In the case of $d=1$, JL03-NFS is exactly same as the Gaussian integer method, and thus parameters B, C are chosen $B = C = L_p[1/2; (1/2)^{1/2}]$ from [2], in particularly $B = C = 5219$ in our experiment. The size of the matrix in the linear algebra step becomes 1318×1318 or 1398×1398 in the minimal or maximal case, respectively.

The parameters d, B, C for Sch06-NFS are chosen based on equations (12) and (13). However the parameters B, C are optimized for the maximal Δ of weight-w primes. Sch06-NFS can be accelerated by choosing smaller B, C if Δ is not maximal for given p. We tried to find the optimal B, C (we set $B = C$) whose running time of Sch06-NFS is smallest among $B_{max} - 1000z$ for $z = 1, 2, ...$, where B_{max} is the starting value defined by equation (13). In our experiment, the smallest or largest size of B, C for 100-bit primes is 6600 or 16600, respectively. The size of the matrix in the linear algebra step becomes 499×499 or 3774×3774 in the minimal or maximal case, respectively. The running time of Sch06-NFS is about 4 times faster than that of choosing $B = C = B_{max}$ in our experiment.

4.3 Comparison between JL03-NFS and Sch06-NFS

We give a comparison of the running time of JL03-NFS and Sch06-NFS using the parameters in Section 4.2 for the same primes.

Figure 1 shows the histogram of the running time of JL03-NFS and Sch06-NFS for the primes of weight less than or equal to five. The total number of such primes is 724 (or 901) for 100 bits (or 110 bits), respectively. The horizontal line presents the running time of JL03-NFS or Sch06-NFS, and the vertical line presents the number of primes whose running time are same in the second scale. The running time of Sch06-NFS has a larger variance than that of JL03-NFS.

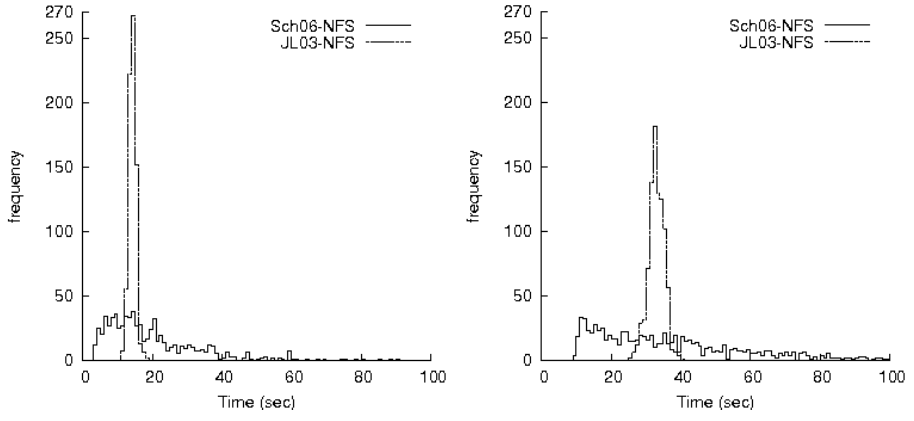


Fig. 1. Timing of Sch06-NFS and JL03-NFS for solving the DLP over GF(p) of low weight less than or equal to 5, where p is 100 bits (or 110 bits) for the figure in the left (or right) side

The main reason is that the coefficients of the definition polynomial f_α of JL03-NFS have similar size for all primes, but those of Sch06-NFS are different for each prime. In the appendix we list some primes, their definition polynomials f_α, f_β, and their speed ratio of JL03-NFS over Sch06-NFS in our experiment.

Next our interest is to know the primes for which Sch06-NFS is faster than JL03-NFS. We call such a prime *the weak prime*. The number of the weak primes in our experiment is about 43% (or 50%) for 100-bit (or 110-bit) primes of low weight less than or equal to 5, respectively. In the largest difference of timing between JL03-NFS and Sch06-NFS using the weak primes, Sch06-NFS is about 3.6 times (or 4.7 times) faster than JL03-NFS for 100 bits (or 110 bits), respectively. Sch06-NFS is asymptotically faster than JL03-NFS, and thus the number of the weak primes and the difference of their running time gradually increase for larger primes.

Table 1. The ratio of weak primes of low weight less than or equal to 5

Bit-length of primes p	# of primes p	# of weak primes	ratio
100	724	310	43%
110	901	454	50%

5 Conclusion

In this paper we presented an experimental result on two versions of number field sieve for solving the discrete logarithm problem over GF(p), namely by Joux-Lercier (JL03-NFS) [4] and Schirokauer for low-weight prime p (Sch06-NFS) [18]. Indeed we presented some comparisons of the running timing of JL03-NFS and Sch06-NFS for primes of 100 and 110 bits with weight less than or equal to 5. In

our experiment on a PC, the variance of timing for Sch06-NFS was larger than that of JL03-NFS, and Sch06-NFS was faster than JL03-NFS for about 43% (or 50%) of primes of 100 bits (or 110 bits), respectively.

Our future work is to have an experiment on the number field sieve over GF(p) with larger prime numbers of low weight using some efficient algorithms such as the lattice sieve and the structured Gaussian method.

Acknowledgements

We thank Yuji Kida for explaining about the decomposition of bad primes, and Kazumaro Aoki for valuable comments on the implementation of the number field sieve.

References

1. Cohen, H.: A Course in Computational Algebraic Number Theory. GTM, vol. 138. Springer, Heidelberg (1995)
2. Coppersmith, D., Odlyzko, A., Schroppel, R.: Discrete Logarithms in GF(p). Algorithmica 1, 1–15 (1986)
3. Gordon, D.: Discrete Logarithms in GF(p) Using the Number Field Sieve. SIAM J. Discrete Math. 6, 124–138 (1993)
4. Joux, A., Lercier, R.: Improvements to the General Number Field Sieve for Discrete Logarithms in Prime Fields. Math. Comp. 72, 953–967 (2003)
5. Kleinjung, T.: On Polynomial Selection for the General Number Field Sieve. Math. Comp. 75, 2037–2047 (2006)
6. Kleinjung, T., et al.: Discrete Logarithms in GF(p) - 160 digits. Posting to the Number Theory List (2007), `http://listserv.nodak.edu/cgi-bin/wa.exe?A2=ind0702&L=nmbrthry&T=0&P=194`
7. LaMacchia, B., Odlyzko, A.: Computation of Discrete Logarithms in Prime Fields. Designs, Codes and Cryptography 1, 47–62 (1991)
8. LaMacchia, B.A., Odlyzko, A.M.: Solving Large Sparse Linear Systems over Finite Fields. In: Menezes, A., Vanstone, S.A. (eds.) CRYPTO 1990. LNCS, vol. 537, pp. 109–133. Springer, Heidelberg (1991)
9. Lanczos, C.: Solution of Systems of Linear Equations by Minimized Iterations. J. Res. Nat. Bur. Stand. 49, 33–53 (1952)
10. Lenstra, A., Lenstra Jr., H.: The Development of the Number Field Sieve. LNM, vol. 1554. Springer, Heidelberg (1993)
11. Lenstra, A., Lenstra Jr, H., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Ann. 261, 513–534 (1982)
12. Lercier, R.: Computations - Discrete Logarithms (2009), `http://perso.univ-rennes1.fr/reynald.lercier/plugins/getfilehtml/getfilehtml7d2c.html?lng=en&id=6`
13. Murphy, B.: Polynomial Selection for the Number Field Sieve Integer Factorisation Algorithm, PhD. thesis, The Australian National University (1999)
14. PARI/GP, version 2.3.4, Bordeaux (2008), `http://pari.math.u-bordeaux.fr/`
15. Pomerance, C., Smith, J.: Reduction of Huge, Sparse Matrices over Finite Fields via Created Catastrophes. Experiment. Math. 1, 89–94 (1992)

16. Schirokauer, O.: Discrete Logarithms and Local Units, Philos. Trans. Roy. Soc. London Ser. A 345, 409–424 (1993)
17. Schirokauer, O.: Virtual Logarithms. J. Algorithms 57, 140–147 (2005)
18. Schirokauer, O.: The Number Field Sieve for Integers of Low Weight. IACR Cryptology ePrint Archive, 2006/107 (2006)
19. Schirokauer, O.: The Number Field Sieve for Integers of Low Weight. Math. Comp. 79, 583–602 (2010)
20. Solinas, J.: Generalized Mersenne Numbers, Technical Report CORR 99-39, University of Waterloo (1999)

A Some Definition Polynomials in Our Experiment

In the following we present some definition polynomials $f_\beta(X)$ and $f_\alpha(X)$ used for JL03-NFS and Sch06-NFS in our experiment. The speed ratio in the below list means the ratio of the running time of JL03-NFS over that of Sch06-NFS for the same prime p of low weight.

prime: $p = 2^{100} - 2^{41} + 2^{37} - 2^8 - 1$, speed ratio: 4.72
JL03-NFS: $f_\beta(X) = X^2 + 7, f_\alpha(X) = -413567362074277X + 265301450084044$
Sch06-NFS: $f_\beta(X) = X - 2^{33}, f_\alpha(X) = 2X^3 - 240X - 257$

prime: $p = 2^{99} + 2^{64} - 2^{36} + 2^4 - 1$, speed ratio 4.32
JL03-NFS: $f_\beta(X) = X^2 + 5, f_\alpha(X) = 502209212339547X + 81118034194511$
Sch06-NFS: $f_\beta(X) = X - 2^{33}, f_\alpha(X) = 4X^3 + X^2 - 32X + 60$

prime: $p = 2^{99} + 2^{93} - 2^{27} - 2^3 - 1$, speed ratio 3.29
JL03-NFS: $f_\beta(X) = X^2 + 5, f_\alpha(X) = -502346890207431X - 160298628663541$
Sch06-NFS: $f_\beta(X) = X - 2^{33}, f_\alpha(X) = 65X^3 - X - 576$

prime: $p = 2^{99} + 2^{56} - 2^{36} + 2^{12} - 1$, speed ratio 1.01
JL03-NFS: $f_\beta(X) = X^2 + 5, f_\alpha(X) = -486590865498129X + 289477543683931$
Sch06-NFS: $f_\beta(X) = X - 2^{33}, f_\alpha(X) = 1024X^3 + X^2 - 8192X + 4193280$

prime: $p = 2^{100} - 2^{68} + 2^{23} + 2^8 - 1$, speed ratio 1.00
JL03-NFS: $f_\beta(X) = X^2 + 5, f_\alpha(X) = -558733761725589X - 987108969367631$
Sch06-NFS: $f_\beta(X) = X - 2^{33}, f_\alpha(X) = 2048X^3 - 4096X^2 + X + 261120$

prime: $p = 2^{100} - 2^{58} - 2^{27} - 2^{12} - 1$, speed ratio 0.99
JL03-NFS: $f_\beta(X) = X^2 + 5, f_\alpha(X) = -814479724965946X + 697184113976221$
Sch06-NFS: $f_\beta(X) = X - 2^{33}, f_\alpha(X) = 512X^3 - X^2 - 4X - 1048832$

prime: $p = 2^{99} + 2^{79} - 2^{29} + 2^{18} - 1$, speed ratio: 0.22
JL03-NFS: $f_\beta(X) = X^2 + 5, f_\alpha(X) = -603314964352898X + 285539809256749$
Sch06-NFS: $f_\beta(X) = X - 2^{33}, f_\alpha(X) = 1048577X^3 - 65504X - 1048576$

prime: $p = 2^{99} + 2^{81} + 2^{56} - 2^{13} - 1$, speed ratio: 0.13
JL03-NFS: $f_\beta(X) = X^2 + 5, f_\alpha(X) = 568615875376675X + 533725664780668$
Sch06-NFS: $f_\beta(X) = X - 2^{33}, f_\alpha(X) = 1048580X^3 + 1024X^2 - X - 1048576$

prime: $p = 2^{99} + 2^{94} - 2^{52} + 2^{10} - 1$, speed ratio 0.11
JL03-NFS: $f_\beta(X) = X^2 + 7, f_\alpha(X) = 42938344745443X + 800453890362416$
Sch06-NFS: $f_\beta(X) = X - 2^{33}, f_\alpha(X) = 8650752X^3 - 512X^2 + X - 8388608$

The Minimum Distance of Graph Codes

Tom Høholdt and Jørn Justesen

[1] Department of Mathematics, The Technical University of Denmark, Bldg. 303 DK 2800 Lyngby Denmark
T.Hoeholdt@mat.dtu.dk
[2] jorn@justesen.info

Abstract. We study codes constructed from graphs where the code symbols are associated with the edges and the symbols connected to a given vertex are restricted to be codewords in a component code. In particular we treat such codes from bipartite expander graphs coming from Euclidean planes and other geometries. We give results on the minimum distances of the codes.

Keywords: Graph codes, Euclidean and projective geometry.

1 Introduction

In 1981 Tanner [1] introduced a construction of error-correcting codes based on graphs and since then a considerable number of results have been obtained [2], [3], [4], [5] and [6]. The recent textbook by Roth [8] contains a thorough presentation of the subject. In this paper we consider some classes of graph codes and in particular codes from bipartite expander graphs based on finite geometries. In this case the vertices of the graph are labeled by the points and lines of a finite geometry, and there is an edge connecting a line vertex to any vertex labeled by a point on the line. The code symbols are associated with the edges, and the symbols connected to a given vertex are restricted to be codewords in a component code over the field that is used for constructing the geometry.

In Section 2 we recall the construction of the codes and we give basic bounds on their parameters. In Section 3 the lower bound on the minimum distance is improved by considering the properties of eigenvectors of the adjacency matrix of the graph. In Section 4 we specialize to bipartite graphs from finite geometries, and in Section 5 we show that the bound obtained is tight for a special class of graph codes. Section 6 contains the conclusion.

2 Basic Parameters and Bounds

We recall the construction of codes based on graphs.

2.1 General n-Regular Graphs

Let $G = (V, E)$ be an n-regular connected graph, without loops and multiple edges, with vertex set V and edge set E. Let $|V| = m_1$, $|E| = \frac{m_1 n}{2} = L$ and

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 201–212, 2011.
© Springer-Verlag Berlin Heidelberg 2011

let C_1 be a (n, k, d) code over the finite field $\mathbb{F}_q$. We now construct a code C of length L over $\mathbb{F}_q$ by associating $\mathbb{F}_q$ symbols with the edges of the graph (in some selected order) and demanding that the symbols connected to a vertex in V shall be a codeword in C_1. It is clear that C is a linear code of length L and if we let K denote the dimension of C we have that $L - K \leq m_1(n - k)$ and therefore

Lemma 1. *The rate* $R = \frac{K}{L}$ *satisfy*

$$R \geq 2r - 1 \text{ , where } r = \frac{k}{n} \text{ is the rate of the component code.}$$

In the following we shall use the *adjacency matrix* matrix of the graph so we recall the definition:

Definition 1. *Let* $z_1, z_2, \ldots, z_{m_1}$ *be the vertices of the graph* G. *The adjacency matrix* $A = (a_{ij}), i, j = 1, 2, \ldots, m_1$ *is defined by*

$$a_{ij} = \begin{cases} 1 \text{ if } z_i \text{ is connected to } z_j \\ 0 \qquad\qquad \text{else} \end{cases}$$

2.2 Bipartite Graphs

With bipartite graphs the construction is as follows. Let $G = (V, E)$ be an n-regular connected bipartite graph, without multiple edges, with vertex set $V = V_1 \cup V_2$ such that $V_1 \cap V_2 = \varnothing$ and $|V_1| = |V_2| = m$. A bipartite graph is n-regular if each vertex of V_1 is connected to n vertices of V_2, and each vertex of V_2 is connected to n vertices of V_1.

Let C_1 be a linear (n, k_1, d_1) code and C_2 a linear (n, k_2, d_2) code both over the finite field $\mathbb{F}_q$. We now construct a code $\mathscr{C}$ of length $L = mn$ over $\mathbb{F}_q$ by associating $\mathbb{F}_q$ symbols with the edges of the graph and demanding that the symbols connected to a vertex of V_1 shall be a codeword of C_1 and that the symbols on the edges connected to a vertex of V_2 shall be a codeword of C_2. More formally we assume an ordering of the edges E of G and for a vertex $u \in V$ let $E(u)$ denote the set of edges incident with u . For a word $\boldsymbol{x} = (x_e)_{e \in E}$ in ${\mathbb{F}_q}^L$ denote by $(\boldsymbol{x})_{E(u)}$ the subword of $\boldsymbol{x}$ that is indexed by $E(u)$, that is $(\boldsymbol{x})_{E(u)} = (x_e)_{e \in E(u)}$. Then the code $\mathscr{C}$ is defined by

$$\mathscr{C} = \{\boldsymbol{c} \in {\mathbb{F}_q}^L : (\boldsymbol{c})_{E(u)} \in C_1 \text{ for every } u \in V_1 \text{ and } (\boldsymbol{c})_{E(u)} \in C_2 \text{ for every } u \in V_2\}$$

It is clear that $\mathscr{C}$ is a linear code. Let K be its dimension. We recall from [3]

Lemma 2. *The rate* $R = \frac{K}{L}$ *of* $\mathscr{C}$ *satisfies*

$$R \geq r_1 + r_2 - 1 \quad \textit{where} \quad r_1 = \frac{k_1}{n} \quad \textit{and} \quad r_2 = \frac{k_2}{n}$$

Proof: The number of linearly independent parity checks is at most $m(n - k_1) + m(n - k_2)$, so $L - K \leq m(n - k_1) + m(n - k_2)$ and since $L = mn$ we get the result. □

Let $x_1, x_2, \ldots, x_m$ be the vertices in V_1 and $y_1, y_2, \ldots, y_m$ the vertices in V_2 and define the $m \times m$ matrix $M = m_{ij}$ by

$$m_{ij} = \begin{cases} 1 \text{ if } x_i \text{ is connected to } y_j \\ 0 \qquad\qquad\quad \text{else} \end{cases}$$

The *adjacency matrix* of the bipartite graph is then

$$A = \begin{pmatrix} 0 & M \\ M^T & 0 \end{pmatrix}$$

2.3 Bounds on the Minimum Distance

In both cases above we have that each row of A has n 1s, the largest eigenvalue of A is n, and the corresponding eigenvector is the all-ones vector. In the bipartite case also $-n$ is an eigenvalue of A, and the corresponding eigenvector has 1s in the first half of the positions and -1 in the rest. It is known [8] that for a connected graph $-n \leq \lambda_i \leq n$ where λ_i is any eigenvalue and that the second largest eigenvalue λ is closely related to the expansion properties of the graph. Large random graphs and known families with good expansion properties have $\lambda = 2\sqrt{n-1}$ [7]. We quote the following bounds on the minimum distances.

Theorem 1. *The minimum distance D of C satisfies*

$$D \geq dm_1 \frac{d-\lambda}{2(n-\lambda)} \tag{1}$$

Theorem 2. *The minimum distance D of $\mathscr{C}$ if $d_1 = d_2 = d$ satisfies*

$$D \geq dm \frac{d-\lambda}{n-\lambda} \tag{2}$$

This bound was obtained by Sipser and Spielman in [6], and a slight modification gives the bound in Theorem 1. For proofs see e.g. [8], Chapter 13.

For a complete n-regular graph, where $m_1 = n+1$, then $\lambda = -1$, the bound of Theorem 1 is

$$D \geq \frac{d(d+1)}{2}$$

which indeed is the right bound for these codes.

We also note that in the case where the bipartite graph is complete, and hence $n = m$ and $\lambda = 0$, we get the usual bound for product codes.

For short component codes, where $d \leq \lambda$, the bound is not useful, but we can get a simple lower bound by the following consideration: Any vertex corresponding to a nonzero codeword on the right side is incident with at least d nonzero edges connecting to vertices in the left set, and these reach at least $d(d-1)$ vertices in the right set with nonzero edges. If the girth of the graph is at least 6, these vertices are distinct, and the minimum distance is always lower bounded by

$$D \geq d(d(d-1)+1) = d(d^2-d+1) \tag{3}$$

If the girth is $g \geq 6$, the argument can be repeated to give

$$D \geq d(1 + d(d-1)\sum_{i=0}^{\frac{g/2-3}{2}}(d-1)^{2i}) \text{ if } g/2 \text{ is odd}$$

$$D \geq d(d\sum_{i=0}^{\frac{g/2-2}{2}}(d-1)^{2i}) \tag{4}$$

if $g/2$ is even. This bound also appears in Skachek's thesis [11].

The potential of graph codes is related to the possibility of keeping the component code fixed while the size of the graph increases. In this way the performance can be improved with only a linear increase in decoding complexity. However, for the codes C and $\mathscr{C}$ to have a reasonable rate, the component codes must have high rate, and to get a positive bound from Theorem 1 and Theorem 2, the minimum distance of the component codes has to be larger than λ. The combination of these requirements tends to make the resulting code too long for any realistic application. Thus our emphasis in this paper is to improve the analysis of codes of moderate block length derived from specific good graphs.

3 Improved Lower Bounds on the Minimum Distance

In Section 5 we demonstrate that the bound of Theorem 2 is actually tight in certain cases. However, in some cases of interest, it is possible to get sharper lower bounds. As a first case we consider component codes of different rates. Even though the resulting bound on the minimum distance for fixed overall rate is maximized by choosing component codes with equal distance (see comment after the proof of Theorem 3), the performance with practical decoding algorithms is improved by using unequal distances (as in the case of product codes). Several generalizations of (2) to unequal distances were presented in [9], and [10].

Theorem 3. *The minimum distance D of $\mathscr{C}$ satisfies*

$$D \geq md_1\frac{d_2 - \lambda\beta}{n - \lambda\beta} \tag{5}$$

where β is the positive root of

$$\beta^2(\alpha n - d_1) + \beta\lambda(1 - \frac{d_1}{d_2}) + d_1 - n = 0$$

Proof: Let $\grave{E}$ be a set of edges in G that supports a nonzero codeword of C. Let S be the subset of vertices in V_1 incident with $\grave{E}$ and let T be the subset of vertices in V_2 incident with $\grave{E}$. We will get the bound on D from a bound of $|\grave{E}|$. We follow the standard line of proof by defining a vector v as a modified indicator vector for the sets S and T, and then apply a well-known result (see e.g. [8], Lemma 13.6)

$$v^T Av \leq \lambda v^T v \tag{6}$$

Equality holds if and only if v is an eigenvector associated with the eigenvalue λ. We obtain improved bounds by adjusting the coordinates of v to values that are consistent with the the properties of an eigenvector.

Suppose that $|S| = a$ and $|T| = \alpha a$, $\alpha \geq 1$, and let e be the average valency of the vertices in S, thus $\frac{e}{\alpha}$ the average valency in T. Let $v = (v_i)$ be a vector of length $2m$ where

$$v_i = \begin{cases} 1 & \text{if } i \in S \\ -\frac{a}{m-a} & \text{if } i \in V_1 \setminus S \\ \beta & \text{if } i \in T \\ -\frac{\alpha\beta a}{m-\alpha a} & \text{if } i \in V_2 \setminus T \end{cases}$$

where $0 < \beta \leq 1$. By balancing v we assure that the inner product of v with the eigenvectors associated with the largest numerical eigenvalue n is 0. Since the multiplicity of n is one it follows that v is in the space spanned by the eigenvectors of A that are associated with the remaining eigenvalues of A. We can directly calculate the left side of (6) since we know that the number of edges connecting S and T is ae, and thus also the number of edges connecting S and $V_2 \setminus T$, namely $(n-e)a$. Therefore the number of edges connecting $V_1 \setminus S$ and T is $a\alpha(n - e/\alpha)$ and the remaining edges connect $V_1 \setminus S$ and $V_2 \setminus T$. We therefore get

$$v^T A v = \tfrac{2ma\beta}{(m-a)(m-\alpha a)}(me - na\alpha)$$

The inequality (6) and this result give

$$\tfrac{2ma\beta}{(m-a)(m-\alpha a)}(me - na\alpha) \leq \lambda(a + (m-a)\tfrac{a^2}{(m-a)^2} + a\alpha\beta^2 + (m-\alpha a)\tfrac{\alpha^2\beta^2a^2}{(m-\alpha a)^2})$$

and this by a straightforward calculation leads to the following bound on a

$$a \geq \frac{m}{\alpha}\frac{2e\beta - \lambda(1+\alpha\beta^2)}{2\beta n - \lambda(1+\beta^2)} \tag{7}$$

which holds for any positive β. The lower bound on a is met if and only if v is an eigenvector associated with the eigenvalue λ, i.e. $Av = \lambda v$, and a necessary condition for this, where we only look at the upper part of A is

$$\lambda = e\beta - (n-e)\frac{a\alpha\beta}{m-a\alpha} \quad \text{and} \quad \beta\lambda = \frac{e}{\alpha} - (n - \frac{e}{\alpha})\frac{a}{m-a}$$

These two conditions lead to the following expressions for a

$$a = m\frac{e\beta - \lambda}{\alpha(\beta n - \lambda)} \tag{8}$$

$$a = m\frac{\frac{e}{\alpha} - \beta\lambda}{n - \beta\lambda} \tag{9}$$

and by eliminating a we get the equation for β. It can be seen that there is a positive solution less than 1.

Maximizing the right side of (7) with respect to β actually leads to the same equation. Thus this is the sharpest lower bound that can be obtained by this method. The bound can be met if there is a subgraph on S and T with exactly valencies e and $\frac{e}{\alpha}$ (which we expect will rarely be the case). Since $D \geq ea$ the lower bound increases with a and e, we thus get a new lower bound by choosing $\alpha = \frac{e}{d_2}$ since the bound on a decreases with α and then choosing $e = d_1$. □

The bound in Theorem 3 improves the bounds obtained in [9] and [10]. They are respectively

$$D \geq \frac{m}{n}(d_1 d_2 - \frac{\lambda}{2}(d_1 + d_2)) \text{ where } d_1 \geq d_2 > \frac{\lambda}{2}.$$

and

$$D \geq m\frac{d_1 d_2 - \lambda\sqrt{d_1 d_2}}{n-\lambda}$$

The comparisons are facilitated by using the approximation $\beta \approx 1/\sqrt{\alpha}$, which can also be used to prove that for fixed rate, i.e. $d_1 + d_2$ fixed, the lower bound is maximum for $d_1 = d_2$.

Example 1. As a case where Theorem 3 gives simple numbers we may take $n = 16, \lambda = 4, d_1 = 8, \alpha = 2$, and consequently $\beta = 2/3$. From (5) we get $D \geq 4m/5$ compared to $m/2$ and $0.78m$ for the earlier bounds.

For Theorem 2 or 3 to hold with equality, the edges connecting vertices in S to $V_2 \setminus T$ must be equally distributed over these vertices (and similarly for edges connecting T to $V_1 \setminus S$). Clearly this is usually not possible because of the integer constraints. In the proof of the following theorem we modify v by distinguishing between the subsets of vertices that are connected to S or T and the remaining vertices. For simplicity we only treat the symmetric case $d_1 = d_2 = d$.

We shall first derive the coordinates of a hypothetical eigenvector corresponding to sets S and T of minimal (equal) size. To get a useful bound for smaller d we denote the set of vertices in $V_2 \setminus T$ that are connected to S as U_2 and the set of vertices in V_2 not in T or U_2 as W_2. Similarly V_1 is divided into S, U_1, and W_1. The eigenvector v' is assumed to have coordinates 1 in positions corresponding to S and T, u in positions corresponding to U_1 and U_2, and w in the remaining positions. We get the smallest value of a by assuming that the $a(n-d)$ edges from S reach distinct vertices in U_2, and that this is consequently the size of the set. It now follows from the assumption that v' is an eigenvector with eigenvalue λ that $u = (\lambda - d)/(n-d)$. Further $|W_1| = f = m - a(n-d+1)$, and since the vector has to be balanced, $w = a(d - \lambda - 1)/f$. Let the number of edges connecting a vertex in U_1 to vertices in W_2 be g. The number of such edges incident with a vertex in W_2, h, then follows. We get the final condition by applying the eigenvalue calculation to a vertex in U_1:

$$1 + (n - g - 1)\frac{\lambda - d}{n - d} + gw = \lambda\frac{\lambda - d}{n - d} \tag{10}$$

The remaining parameter in v' should be selected to minimize a for a given value of m. The minimum is always on the boundary of the range $0 \leq g \leq n - 1$ and

$0 \leq h \leq n$. The condition $h = n$ applies for $\lambda + 1 < d$ down to a value close to $d = \lambda$. For smaller d the limit is $g = n - 1$. Both conditions hold when

$$\lambda^2 + \lambda(n - d) - d(n - 1) = 0 \tag{11}$$

which clearly has a solution $\lambda = d - \epsilon$ for a small positive ϵ.

From the properties of such a potential eigenvector we get the following lower bound:

Theorem 4. *The minimum distance of C is lower bounded by $D \geq da$ where*

$$m/a \leq 1 + n - d + \frac{(n-1)(n-d)(\lambda - d + 1)}{n - d - \lambda^2 + d\lambda} \tag{12}$$

for $(\lambda^2 - n)/(\lambda - 1) < d \leq \lambda + \epsilon$ and

$$m/a \leq 1 + n - d + \frac{(n-d)\lambda(\lambda - d + 1)}{n(d - \lambda)} \tag{13}$$

for $\lambda + 1 \geq d \geq \lambda + \epsilon$, where ϵ is a positive number derived below.

Proof: The expression (13) in the Theorem follows from (10). However to arrive at a solution with positive parameters in the other case we must assume $(\lambda^2 - n)/(\lambda - 1) < d$, which also ensures that the denominator in (12) is positive. To prove that the eigenvectors give actual lower bounds on the minimum distance of the code we assume that a minimum weight codeword defines S and T as before. We then construct the vector v using the value u from the eigenvector and choose the value w' of the remaining coordinates to get a balanced vector. For $d > \lambda$ we minimize the number of additional vertices, f, by letting each have $h = n$ connections to W_2. The result then follows by choosing w to get a balanced vector. For $d = \lambda$ we get $u = 0$, and from $h = n$ we directly get $m/a \leq 2n - d$. For $d < \lambda$ the vector v is inserted in the inequality (5), we get an inequality for a/m which depends on the parameter corresponding to g. The minimum is again always on the boundary. Thus the only remaining variable is a. Calculating the two sides of (5) we find

$$v^T A v = ad + 2a(\lambda - d) + a(n - g - 1)(\lambda - d)^2/(n - d) + 2ag(\lambda - d)w + f(n - h)w^2$$

and

$$\lambda v^T v = \lambda a + \lambda a(\lambda - d)^2/(n - d) + \lambda f w^2.$$

The terms containing a factor w or fw^2 vanish for small a/m. We can then reduce the inequality by the factor $(\lambda - d)$, which is positive. We then find that the inequality (5) cannot be satisfied for very small a as long as $(\lambda^2 - n)/(\lambda - 1) < d$, and consequently A cannot have eigenvalue λ. The smallest value of a that lets (5) be satisfied gives equality and thus v is the eigenvector. □

Theorem 4 improves on (2) and (3) when the graph is large and d is close to λ as demonstrated in Example 4. As a case of particular interest we mention that for $d = \lambda$, $D \geq dm/(2n - d)$. For very small d, the approach could be extended by considering additional subsets of vertices (reached from S and T in several steps), but the improvements would only apply to very long codes.

For a general n-regular graph we similarly split the set of vertices into S, U, and W, and the same derivation gives lower bounds on the minimum distance that are half the values of (12) and (13).

4 Expander Graphs from Geometries

Certain bipartite graphs derived from generalized polygons have good expansion properties [4], and hence the codes derived from these have large minimum distances. The generalized polygons are incidence structures consisting of points and lines where any point is incident with the same number of lines, and any line is incident with the same number of points. A generalized N-gon, where N is a natural number, defines a bipartite graph $G = (V, E)$ that satisfies the following conditions:

- For all vertices $u, v \in V$, $d(u, v) \leq N$, where $d(u, v)$ is the length of the minimum path connecting u and v.
- If $d(u, v) = s < N$, then there is a unique path of length s connecting u and v.
- Given a vertex $u \in V$ there exists a vertex $v \in V$ such that $d(u, v) = N$.

We note that this implies that the girth of the bipartite graph is at least $2N$. Most of this paper is concerned with graphs from finite planes, and in this context the 3-gons are derived from finite projective planes. (The definition and properties of these can be found in [14], Chapter 2).

Let M be an incidence matrix for a projective plane over $\mathbb{F}_q$ with $m = q^2+q+1$ points with homogeneous coordinates $(x : y : z)$ and $q^2 + q + 1$ lines with homogeneous coordinates $(a : b : c)$ where a point is incident with a line if $ax + by + cz = 0$. The bipartite graph then has adjacency matrix

$$A = \begin{pmatrix} 0 & M \\ M^T & 0 \end{pmatrix}$$

The graph is invariant to an interchange of the two sets of variables $(x : y : z)$ and $(a : b : c)$.

Thus each row of A has $q + 1$ 1s so the largest eigenvalue is $q + 1$ and the corresponding eigenvector is the all-ones vector. The graph may be seen as a simple expander graph: The eigenvalues are $\pm(q + 1)$ and $\pm\sqrt{q}$ (all real since A is symmetric). (See [4].)

Starting from a vertex in the right set, $q + 1$ vertices in the left set can be reached in one transition, and $q(q + 1)$ vertices in the right set can be reached from these vertices. The graph can be used to define a code by associating a symbol with each edge and letting all edges that meet in a vertex satisfy the

parity checks of an (n, k, d) code, where $n = q + 1$. Thus the length of the total code is

$$L = mn = (q^2 + q + 1)(q + 1)$$

It is sometimes more convenient to let M be an incidence matrix for an Euclidean plane [14] with $m = q^2$ points, (x, y), and q^2 lines of the form $y = ax + b$. The lines of the form $x = c$ are omitted, and in this way the graph is invariant to an interchange of the two sets of variables.

Thus each row of the adjacency matrix has q 1s and the eigenvalues are $\pm q$, $\pm\sqrt{q}$ and 0 [4].

All edges that meet in a vertex satisfy the parity checks of an (n, k, d) code with $n = q$. Thus the length of the code is

$$L = q^3$$

Example 2. For $q = 4$, the projective plane and $(5, 3, 3)$ component codes give codes of length $L = 5 \cdot 21 = 105$. The minimum distance is lower bounded by (2) and (3), which in this case give the same value

$$D \geq 21 \cdot 5 \cdot (d - 2)/(n - 2) = 21$$

A subgraph with 7 vertices of degree 3 on each side can be found as a binary sub-plane, and for this reason the lower bound is tight. The rate is lower bounded by $R \geq 2 \cdot 3/5 - 1 = 1/5$, but later we shall see that the actual dimension is 29. If the vertices are labeled $(x : y : z)$ and $(a : b : c)$ where the last nonzero coordinate is chosen to be 1, the vertex $(\alpha : 1 : 1)$ is connected to the 5 vertices $(\alpha^2 : 1 : 0), (\alpha^2 : 0 : 1), (0 : 1 : 1), (\alpha : \alpha : 1), (1 : \alpha^2 : 1)$. We can find a basis for the component codewords by evaluating z^2, yz, and y^2.(See e.g. [16] p. 69). Thus the generator matrix of the component code becomes

$$G = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & \alpha & \alpha^2 \\ 1 & 0 & 1 & \alpha^2 & \alpha \end{pmatrix}$$

In particular the codeword $(1, 1, 1, 0, 0)$ is part of the binary sub-plane.

It is possible to construct longer codes from generalized N-gons, but it is known [12] that for $N > 6$ there are no N-gons with degree $q + 1$.

5 Minimum Distances of Codes from Geometries

For the specific codes constructed from graphs derived from finite geometries it is possible to get tighter bounds on the minimum distances, and in some cases we can determine the exact value. Such results provide some insight into the structure of the code and the tightness of the bounds. The use of RS component codes also serves to allow a combination of good rates and distances for moderate code lengths.

When $q = 2^r$, the field $\mathbb{F}_q$ contains a subfield with $q' = 2^s$ symbols whenever s divides r. With the chosen coordinates for the projective plane, the component extended RS code has $q' + 1$ positions with coordinates in the subfield. If the minimum distance of the component code is $q' + 1$, it has a codeword which is 1 in these positions and zero otherwise.

The projective plane contains a subfield projective plane over $\mathbb{F}_{q'}$. With our choice of coordinates, such a plane may be found by taking the vertices that have coordinates in the subfield and the edges incident to these vertices. It now follows from the remark above that if $q' + 1$ is the minimum distance of the component code, the graph code has a codeword which is 1 on the edges corresponding to the subfield plane and zero otherwise.

Since $\mathbb{F}_2$ is a subfield of any field of characteristic 2, there is always a subplane with 7 points and lines, and thus for $d = 3$, the minimum distance of the graph code is ≤ 21. In this case the lower bound (3) is satisfied with equality and 21 is the actual minimum distance. Similarly the lower bound

$$(q' + 1)(q'^2 + q' + 1)$$

is reached by a codeword on the sub-plane whenever the component code has $d = q' + 1$.

For $q = 2^{2r}$, $\mathbb{F}_{2^r}$ is a subfield, and in this case the codeword in the subfield plane has weight satisfying both of the bounds (2) and (3). Thus it is seen that this is the case where the two bounds coincide. Actually we have the more general result:

Theorem 5. *For $q = 2^{2r}$ and any d, $2^r + 1 \leq d \leq 2^{2r}$ there is a graph code with generalized RS component codes such that the minimum distance D satisfies Theorem 2 with equality.*

Proof: It is well known that we can order the points of the projective plane in a cyclic way as powers of a non-primitive element of $\mathbb{F}_{q^3}$. Similarly, within this sequence the powers of an element of order $2^{2r}+2^r+1$ are the points in a subfield plane (although these are not the points that have subfield coordinates). The cosets of this cyclic subgroup are other versions of the smaller projective plane. It follows that each line in the original plane is a line in one of the subplanes and has exactly one point in each of the other subplanes. Thus by combining the required number of these cosets we can get graphs of any required degree. By assigning symbols to the edges and choosing the appropriate scaling of the symbols in the component codes, we get a codeword with the weight indicated by Theorem 2. □

Example 3. For $q = 16$, the projective plane and component codes of length 17 give codes of length $L = 4641$. The minimum distance is lower bounded by

$$D \geq 21d(d - 4)$$

On each side of the graph, the vertices can be divided into a set of 21 vertices corresponding to the points of a subplane over $q = 4$, and 12 shifts of this set.

From unions of such sets we can construct the balanced eigenvectors needed for the lower bound on the minimum distance to be tight. Thus at least for some choice of the mapping of component code symbols on the edges of the graph, the lower bound is tight for $d \geq 5$.

In the Euclidean plane, we get a slightly higher value of the bounds for $d = \sqrt{q}+1$, but (2) does not give an integer value. The configuration of $d^2 - d + 1$ points and lines, which support minimum weight codewords in the projective planes, do not exist in Euclidean planes. Thus for $d = \sqrt{q}$ we get $a \geq q - \sqrt{q} + 1$, but the bound is not tight. However, there may be codewords supported by the $q-1$ nonzero points of a subplane. Theorem 4 gives $a = \frac{m}{2n-d} = \frac{q^2}{2q-\sqrt{q}}$, which is clearly weaker in this case.

Bipartite graphs derived from generalized quadrangles produce longer codes from small component codes. Thus the bound of Theorem 4 may be of interest for such codes.

Example 4. Consider the generalized quadrangle over F_8. In this case there are 585 nodes on each side of the graph. The second eigenvalue is $\sqrt{2q} = 4$. For $d = 3$, the bound (4) gives at least 15 nonzero vertices, and a codeword of this weight can be constructed by taking the F_2 subset of the graph. For $d = 4$ the same bound gives 40 vertices, but from Theorem 4with $n = 9$ we find that at least $\lceil m/14 \rceil = 42$ vertices are nonzero. In this case the integer constraints are not directly satisfied, and a corresponding eigenvector cannot exist, whereas with $a = 45$ it may be possible to get a construction similar to that in the proof of Theorem 4 with $|W| = 7a$.

6 Conclusion

We have derived a new bound on the minimum distance of some graph codes and have analyzed some of these when the underlying graph comes from a finite geometry.

Acknowledgement

This work was supported in part by Danish Resarch Council Grant 272-07-0266.

References

1. Tanner, M.: A Recursive Approach to Low Complexity Codes. IEEE Trans. Inform. Theory 27, 533–547 (1981)
2. Zémor, G.: On expander codes. IEEE Trans. Inform. Theory (Special Issue on Codes on Graphs and iterative Algorithms) 47, 835–837 (2001)
3. Barg, A., Zémor, G.: Error exponents of expander codes. IEEE Trans. Inform. Theory 48, 1725–1729 (2002)

4. Tanner, M.: Explicit Concentrators from Generalized N-Gons. SIAM J. Alg. Disc. Meth. 5(3), 287–293 (1984)
5. Tanner, M.: Minimum-Distance Bounds by Graph Analysis. IEEE Trans. Inform. Theory 47, 808–821 (2001)
6. Sipser, M., Spielman, D.A.: Expander Codes. IEEE Trans. Inform. Theory 42(6), 1710–1722 (1996)
7. Davidoff, G., Sarnak, P., Valette, A.: Elementary Number Theory, Group Theory, and Ramanujan Graphs, vol. 55. London Mathematical Society, Student Texts (2003)
8. Roth, R.M.: Introduction to Coding Theory. Cambridge University Press, Cambridge (2006)
9. Janwa, H., Lal, A.K.: On Tanner Codes: Minimum Distance and Decoding. AAECC 13, 335–347 (2003)
10. Roth, R.M., Skachek, V.: Improved Nearly-MDS Expander Codes. IEEE Trans. Inform. Theory 52(8), 3650–3661 (2006)
11. Skachek, V.: Low-Density-Parity-Check Codes: Constructions and Bounds, Ph.D. Thesis, Technion, Haifa, Israel (January 2007)
12. Feit, W., Higman, G.: The nonexistence of certain generalized polygons. J. Algebra 1, 114–131 (1964)
13. van Maldeghem, H.: Generalized Polygons. Birkhäuser-Verlag, Basel (1998)
14. Blake, I.A., Mullin, R.C.: The Mathematical Theory of Coding. Academic Press, New York (1975)
15. Lauritzen, N.: Concrete Abstract Algebra. Cambridge University Press, Cambridge (2005)
16. Blahut, R.E.: Algebraic Codes on Lines, Planes, and Curves. Cambridge University Press, Cambridge (2008)
17. Høholdt, T., Justesen, J.: Graph codes with Reed-Solomon component codes. In: Proceedings ISIT 2006, Seattle, Washington, pp. 2022–2026 (July 2006)

Local Duality and the Discrete Logarithm Problem

Ming-Deh Huang

Department of Computer Science, University of Southern California,
Los Angeles, CA 90089-0781, USA

Abstract. It is shown that the computational complexity of Tate local duality is closely related to that of the discrete logarithm problem over finite fields. Local duality in the multiplicative case and the case of Jacobians of curves over p-adic local fields are considered. When the local field contains the necessary roots of unity, the case of curves over local fields is polynomial time reducible to the multiplicative case, and the multiplicative case is polynomial time equivalent to computing discrete logarithm in finite fields. When the local field dose not contains the necessary roots of unity, similar results can be obtained at the cost of going to an extension that does contain these roots of unity.

1 Introduction

In this paper we study the computational complexity of Tate local duality in the multiplicative case and in the case of Jacobians of curves over p-adic local fields. We show that when the local field contains the necessary roots of unity, the case of curves over local fields is polynomial time reducible to the multiplicative case, and the multiplicative case is polynomial time equivalent to computing discrete logarithm in finite fields. In this case the pairing objects can be efficiently represented and except for one discrete logarithm computation, the entire computation of local duality can be made rational over the local field and carried out efficiently in the residue finite field. When the local field dose not contains the necessary roots of unity, it is unclear whether the same statement is true, even though similar results can be obtained at the cost of going to an extension that does contain these roots of unity.

Throughout the paper k will be a p-adic local field with a residue field $\mathbb{F}$, k^s a fixed separable closure of k, and $G_k = \mathrm{Gal}(k^s/k)$ the absolute Galois group over k. Let k^{ur} be the maximal unramified subfield of k^s, let $\mathcal{I}$ denote the inertia group $\mathrm{Gal}(k^s/k^{ur})$, and $G_{\mathbb{F}} = \mathrm{Gal}(k^{ur}/k) \cong \mathrm{Gal}(\bar{\mathbb{F}}/\mathbb{F})$ where $\bar{\mathbb{F}}$ denotes an algebraic closure of $\mathbb{F}$. Let v denote the unique discrete valuation of k and π be a uniformizing element. Let m be a natural number not divisible by p, the characteristic of $\mathbb{F}$. Let $\mu_m(\bar{\mathbb{F}})$ and $\mu_m(k^s)$ denote the group of m-th roots of unity in $\bar{F}$ and k^s respectively. We will write μ_m instead of $\mu_m(\bar{\mathbb{F}})$ or $\mu_m(k^s)$ when the context is clear. When $\mu_m \subset k$, let $k_{\pi,m}$ denote the field $k(\pi^{1/m})$ where $\pi^{1/m}$ is an m-th root of π in k^s.

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 213–222, 2011.
© Springer-Verlag Berlin Heidelberg 2011

1.1 The Multiplicative Case

There is a non-degenerate pairing:

$$\langle , \rangle : H^1(k, \mathbb{Z}/m\mathbb{Z}) \times k^*/k^{*m} \to Br(k)[m] \stackrel{\text{inv}}{\to} \frac{1}{m}\mathbb{Z}/\mathbb{Z} \cong \mathbb{Z}/m\mathbb{Z}$$

which we refer to as the (Tate) local duality in the multiplicative case (see [6] I§ 2).

Let G_{ab} denote the Galois group of the maximal abelian extension of k. Then $H^1(k, \mathbb{Z}/m\mathbb{Z}) \cong \text{Hom}(G_{ab}, \mathbb{Z}/m\mathbb{Z})$. For $\chi \in H^1(k, \mathbb{Z}/m\mathbb{Z})$ and $\alpha \in k^*$,

$$\langle \chi, \alpha \rangle = \chi(\theta(\alpha))$$

where $\theta : k^* \to G_{ab}$ is the local Artin map. From local class field theory we also have that

$$\chi(\theta(\alpha)) = \text{inv}(\alpha \cup \delta\chi)$$

where $\alpha \in k^* = H^0(k, k^{s*})$, $\delta\chi$ is the image of $\chi \in H^1(k, \mathbb{Z}/m\mathbb{Z}) \cong H^1(k, \frac{1}{m}\mathbb{Z}/\mathbb{Z}) \subset H^1(k, \mathbb{Q}/\mathbb{Z})$ in the connecting map $H^1(k, \mathbb{Q}/\mathbb{Z}) \stackrel{\delta}{\to} H^2(k, \mathbb{Z})$ with respect to

$$0 \to \mathbb{Z} \to \mathbb{Q} \to \mathbb{Q}/\mathbb{Z} \to 0$$

and $\text{inv} : \text{Br}(k) \to \mathbb{Q}/\mathbb{Z}$ is the invariant map (See [1] VI, [8] XI §3, [6] I §1).

We discuss how elements in the pairing groups can be represented for purpose of computation. Each element of k^*/k^{*m} can be specified in the form $a\pi^i$ with $a \in \mathbb{F}^*/\mathbb{F}^{*m}$ and $0 \le i \le m-1$. This is because

$$k^*/k^{*m} \cong \mathbb{F}^*/\mathbb{F}^{*m} \times \{\pi^i \mid i = 0, ..., m-1\}.$$

Since $H^1(k, \mathbb{Z}/m\mathbb{Z}) \cong \text{Hom}(G_{ab}, \mathbb{Z}/m\mathbb{Z})$, each $\chi \in H^1(k, \mathbb{Z}/m\mathbb{Z})$ is determined by the cyclic extension of degree dividing m fixed by $\ker\chi$, and $\chi(\sigma)$ where σ is a generator of $G_{ab}/\ker\chi$.

We will primarily deal with the case that the m-th roots of unity are contained in k. This is equivalent to saying that $m \mid q-1$ where $\#\mathbb{F} = q$. The case that k does not contain all m-th roots of unity will be discussed in §4.

Since k contains m-roots of unity, the field fixed by $\ker\chi$ is generated by an m-th root of some $\alpha \in k^*$. If we fix a primitive m-th root of unity ζ in k (represented by an m-th root of unity in $\mathbb{F}^*$), then since $H^1(k, \mathbb{Z}/m\mathbb{Z}) \cong \text{Hom}(G_{ab}, \mathbb{Z}/m\mathbb{Z})$, an element $\chi \in H^1(k, \mathbb{Z}/m\mathbb{Z})$ can be specified by an element $\alpha \in k^*/k^{*m}$ where $k(\alpha^{1/m})$ is the field fixed by $\ker\chi$ and for $\sigma \in G_{ab}$, $\chi(\sigma) = i$ if $\sigma(\alpha^{1/m}) = \zeta^i\alpha^{1/m}$. And as discussed earlier, each $\alpha \in k^*/k^{*m}$ can be represented by an element in $\mathbb{F}^*/\mathbb{F}^{*m} \times \{0, \pi, ..., \pi^{m-1}\}$. Under this representation we prove the following:

Theorem 1. *Suppose $\mu_m \subset k$. The discrete-log problem on the subgroup of order m of $\mathbb{F}$ is polynomial time solvable if and only if the local duality*

$$H^1(k, \mathbb{Z}/m\mathbb{Z}) \times k^*/k^{*m} \to \mathbb{Z}/m\mathbb{Z}$$

is polynomial time computable.

1.2 The Case of Jacobians of Curves

Let A be a principally polarized abelian variety over k. For any field K containing k let $A(K)$ denote the group of K-rational points on A. Let $A[m] = A(k^s)[m]$. We have a non-degenerate pairing:

$$H^1(k, A)[m] \times A(k)/mA(k) \to \mathrm{Br}(k)[m] \stackrel{\mathrm{inv}}{\to} \frac{1}{m}\mathbb{Z}/\mathbb{Z} \cong \mathbb{Z}/m\mathbb{Z},$$

which is the Tate local duality for abelian varieties over the local field k (See [6] I § 3, [10]). This pairing can be described as follows. By taking Galois cohomology from the exact Kummer sequence of G_k-modules:

$$0 \to A[m] \to A(k^s) \stackrel{m}{\to} A(k^s) \to 0$$

we get the exact sequence:

$$0 \to A(k)/mA(k) \stackrel{\delta}{\to} H^1(k, A[m]) \to H^1(k, A)[m] \to 0.$$

For $\alpha \in H^1(K, A)[m]$ and $R \in A(k)$, the pairing between α and R is defined to be $\mathrm{inv}(\delta R \cup \beta)$ where $\beta \in H^1(k, A[m])$ is such that its image is α in $H^1(k, A[m]) \to H^1(k, A)[m]$, and the cup product is

$$H^1(k, A[m]) \times H^1(k, A[m]) \to H^2(k, \mu_m) = \mathrm{Br}(k)[m]$$

relative to the Weil pairing

$$A[m] \times A[m] \to \mu_m$$

where $A[m]$ is identified with the m-torsion group of the dual abelian variety of A through a canonical principal polarization.

We discuss how elements in $H^1(k, A)[m]$ and $A(k)/mA(k)$ can be efficiently represented. We assume that A has good reduction at v. For now we assume $\mu_m \subset k$. The case where $\mu_m \not\subset k$ will be discussed in §4.

Since p does not divide m, and A has good reduction at v, $A(k)/mA(k)$ is isomorphic to $\tilde{A}(\mathbb{F})/m\tilde{A}(\mathbb{F})$ through the reduction map, where $\tilde{A}$ denote the reduction of A at v. Hence an element of $A(k)/mA(k)$ can be represented by its reduction in $\tilde{A}(\mathbb{F})/m\tilde{A}(\mathbb{F})$.

Since $\mu_m \subset k$, $\mathrm{Gal}(k_{\pi,m}/k)$ is cyclic of order m. Let τ be a generator of $\mathrm{Gal}(k_{\pi,m}/k)$ such that $\tau(\pi^{1/m})/\pi^{1/m} = \zeta$. Since $\mu_m \subset k$ and A has good reduction at v, $H^1(k, A)[m] = \mathrm{Hom}(< \tau >, A(k)[m])$ (see Lemma 2.2 of [3], or Lemma 10.1.2 of [7]). An element $f \in \mathrm{Hom}(< \tau >, A(k)[m])$ can be represented by $f(\tau) \in A(k)[m]$.

We are interested in the case of Jacobians of curves. In this case we will use a variant of the Tate pairing defined by Lichtenbaum [3], [5], which turns out to be identical to the Tate pairing up to a sign.

Let C be a smooth projective irreducible curve over k of genus greater than 0, with a k-rational point. We assume that C has good reduction at v, denoted by $\tilde{C}$.

Let $\mathrm{Div}^0(C)$ denote the group of divisors of C of degree 0 over k^s, $\mathcal{P}(C)$ the group of principal divisors of C over k^s, and $\mathrm{Pic}^0(C)$ the factor group $\mathrm{Div}^0(C)/\mathcal{P}(C)$, which is isomorphic to $J(k^s)$ where J is the Jacobian variety of C over k. More generally for a field k' with $k \subset k' \subset k^s$, let $\mathrm{Div}^0_{k'}(C)$ denote the group of divisors of C of degree 0 over k', $\mathcal{P}_{k'}(C)$ the group of principal divisors of C over k', and $\mathrm{Pic}^0_{k'}(C)$ the factor group $\mathrm{Div}^0_{k'}(C)/\mathcal{P}_{k'}(C)$, which is isomorphic to $J(k')$ where J is the Jacobian variety of C over k. These groups are naturally G_k-modules. Lichtenbaum's pairing can be described as follows. By taking Galois cohomology from the exact sequence

$$0 \to \mathcal{P}(C) \to \mathrm{Div}^0(C) \to \mathrm{Pic}^0(C) \to 0$$

we get

$$0 = H^1(k, \mathrm{Div}^0(C)) \to H^1(k, \mathrm{Pic}^0(C)) \xrightarrow{\delta} H^2(k, \mathcal{P}(C)).$$

For $\alpha \in H^1(k, \mathrm{Pic}^0(C))$ and $\bar{D} \in \mathrm{Pic}^0_k(C)$, the pairing of α and $\bar{D}$ is

$$(\alpha, \bar{D}) = \mathrm{inv}[(f_{\sigma,\tau}(D))_{\sigma,\tau \in G_k}]$$

where $\delta\alpha = [(f_{\sigma,\tau})_{\sigma,\tau \in G_k}]$ with $f_{\sigma,\tau} \in k^s(C)$ and $D \in \bar{D}$ such that D is prime to the principal divisors $(f_{\sigma,\tau})$ for all $\sigma, \tau \in G_k$.

Theorem 2. *Suppose $\mu_m \subset k$. Let C be a smooth projective irreducible curve over k of genus greater than 0, with a k-rational point, and good reduction at v. Let m be a positive integer prime to p, the characteristic of $\mathbb{F}$. Suppose k contains a primitive m-th root of unity denoted by ζ. Let τ be a generator of $Gal(k_{\pi,m}/k)$ such that $\tau(\pi^{1/m})/\pi^{1/m} = \zeta$. Let $\chi \in H^1(G_k, \mathbb{Q}/\mathbb{Z})$ be the composition of the natural homomorphism from G_k to $Gal(k_{\pi,m}/k)$ and the homomorphism from $Gal(k_{\pi,m}/k)$ to $\mathbb{Q}/\mathbb{Z}$ that sends τ to $\frac{1}{m}$. Let $\alpha \in H^1(k, Pic^0(C))[m]$ and suppose when identifying $H^1(k, Pic^0(C))[m]$ with $Hom(< \tau >, Pic^0_k(C)[m])$ we have $\bar{S} = \alpha(\tau)$. Let $\bar{D} \in Pic^0_k(C)$. Then*

$$(\alpha, \bar{D}) = \langle \chi, F_S(D) \rangle$$

where F_S is a function in $k(C)$ such that $(F_S) = mS$ with $S \in \bar{S}$, and $D \in \bar{D}$ is such that D is prime to S.

For computation $F_S(D)$ is represented by $\tilde{F}_{\tilde{S}}(\tilde{D})$ where $\tilde{S}$, $\tilde{D}$, and $\tilde{F}_{\tilde{S}}$ is the reduction at v of S, D, and F_S. In fact $\tilde{F}_{\tilde{S}}(\tilde{D})$ is the value of the *Tate-Lichtenbaum pairing*:

$$\mathrm{Pic}^0_{\mathbb{F}}(\tilde{C})[m] \times \mathrm{Pic}^0_{\mathbb{F}}(\tilde{C})/m\mathrm{Pic}^0_{\mathbb{F}}(\tilde{C}) \to \mathbb{F}^*/\mathbb{F}^{*m}$$

defined in [3]. This pairing is well known in cryptography [2],[4], and it can be computed in polynomially many group operations in $\mathrm{Pic}^0_{\mathbb{F}}(\tilde{C})$. From this and the proof of Theorem 1 we will have the following:

Theorem 3. *Suppose $\mu_m \subset k$. Then the local duality*

$$H^1(k, Pic^0(C)) \times Pic^O_k(C)/mPic^0_k(C) \to \mathbb{Z}/m\mathbb{Z}$$

is computable with polynomially many group operations in $Pic^0_{\mathbb{F}}(\tilde{C})$ and solving one discrete logarithm problem in the subgroup of order m of $\mathbb{F}$.

The above theorems show that when $\mu_m \subset k$, local duality in the multiplicative case and the case of Jacobians of curves over local fields can both be computed in polynomially many group operations over $\mathbb{F}$, together with solving a discrete logarithm problem in $\mathbb{F}$. The discrete logarithm problem in $\mathbb{F}$ can be solved in subexponential time [9]. Therefore local duality in the multiplicative can be computed in subexponential time. And local duality in the curve case can be computed in subexponential time, if the group law in the Jacobian of the curve is efficiently computable.

When $\mu_m \not\subset k$, similar results can be obtained at the cost of going up to the extension $\mathbb{F}(\mu_m)$. These results are stated and proven in §4. An interesting open question is whether similar results can be obtained without taking the extension. This question appears to be very difficult, since a positive answer will likely yield a subexponential time algorithm for solving the elliptic curve discrete logarithm problem.

2 Proof of Theorem 1 – The Multiplicative Case

In Theorem 1 we assume that k contains a primitive m-th root of unity ζ. For $a, b \in k^*$, the (local) *norm residue symbol* [1] is defined by:

$$(a, b)_v = \frac{\theta(b)\alpha}{\alpha}$$

where $\alpha^m = a$. The proof of Theorem 1 essentially boils down to verifying that the norm residue symbol is polynomial time computable.

Let b be an element of k^*/k^{*m} represented by $(\tilde{b}, \pi^i)$ in $\mathbb{F}^*/\mathbb{F}^{*m} \times \{\pi^i \mid i = 0, ..., m-1\}$. Let χ be an element of $H^1(k, \mathbb{Z}/m\mathbb{Z})$ specified by an element $a \in k^*/k^{*m}$, where $k(\alpha^{1/m})$ is the field fixed by $\ker \chi$ and for $\sigma \in G_{ab}$, $\chi(\sigma) = i$ if $\sigma(\alpha) = \zeta^i \alpha$, where $\alpha^m = a$. (The element a is in turn represented by an element of $\mathbb{F}^*/\mathbb{F}^{*m} \times \{\pi^i \mid i = 0, ..., m-1\}$.)

From the definition of χ and the fact that $\langle \chi, b \rangle = \chi(\theta(b))$ we have

$$\langle \chi, b \rangle = i \Leftrightarrow \theta(b)\alpha = \zeta^i \alpha \Leftrightarrow (a, b)_v = \zeta^i.$$

So $\langle \chi, b \rangle$ can be obtained from $(a, b)_v$ by taking the discrete logarithm based ζ. Therefore the theorem follows if the norm residue symbol $(a, b)_v$ is computable in polynomial time. This follows from Prop. 8 in XIV, §3 of [8], or a simple derivation which we provide below.

Suppose a is a unit. Then $k(\alpha)$ is an unramified abelian extension over k and $\theta(b)$ when restricted to $k(\alpha)$ is $\tau^v(b)$ where $\tau \in \mathrm{Gal}(k(\alpha)/k)$ is the Frobenius automorphism. Let u be the extension of v to $k(\alpha)$. Then

$$\tau\alpha \equiv \alpha^q \mod u$$

where $q = \#\mathbb{F}$ and since α is a unit

$$\tau\alpha/\alpha \equiv \alpha^{q-1} \equiv a^{\frac{q-1}{m}} \mod u.$$

Hence

$$(a,b)_v = (\tau\alpha/\alpha)^{v(b)}$$

and can be represented by $\tilde{a}^{\frac{q-1}{m}v(b)} \in \mu_m(\mathbb{F})$. So in this case $(a,b)_v$ is computable in polynomial time.

Since for $a, b \in k^*$, $(a,b)_v(b,a)_v = 1$ ([1] p.351), it follows that $(a,b)_v$ is computable in polynomial time if either a or b is a unit.

Since $(a,-a)_v = 1$ for all $a \in k^*$ ([1] p. 350), we have $(\pi,-\pi) = 1$. Since

$$(-1,\pi)_v = (-1)^{\frac{q-1}{m}v(\pi)} = (-1)^{\frac{q-1}{m}},$$

it follows that $(\pi,-1)_v = (-1)^{\frac{q-1}{m}}$, so

$$(\pi,\pi)_v = (\pi,-\pi)_v(\pi,-1)_v = (-1)^{\frac{q-1}{m}}.$$

Therefore in all cases $(a,b)_v$ can be computed in polynomial time, and Theorem 1 follows.

3 Proof of Theorem 2 – The Case of Jacobians of Curves

The proof of the theorem involves cohomological computations. Recall that in defining Galois cohomology over G-modules we can take the resolution P of $\mathbb{Z}$:

$$... \to P_1 \to P_0 \to \mathbb{Z} \to 0$$

where $P_i = \mathbb{Z}[G^{i+1}]$, and form the complex $K(A) = \mathrm{Hom}_G(P, A)$ for a G-module A. Then $H^i(G,A)$ is the i-th cohomology group of this complex. An element of $K^i(A) = \mathrm{Hom}_G(P_i, A)$ is determined by a function from G^i to A. Let d denote the boundary maps

$$...K^i(A) \xrightarrow{d} K^{i+1}(A)....$$

For an exact sequence of G-modules

$$0 \to A \to B \to C \to 0$$

the induced

$$0 \to K^i(A) \to K^i(B) \to K^i(C) \to 0$$

is also exact, and we let δ denote the connecting homomorphisms

$$\delta : H^i(G,C) \to H^{i+1}(G,A).$$

Let $\tau \in \mathrm{Gal}(k(\pi^{\frac{1}{m}})/k)$ be such that $\tau\pi^{\frac{1}{m}} = \pi^{\frac{1}{m}}\zeta$. Let $\chi \in H^1(G_k, \mathbb{Q}/\mathbb{Z})$ be the composition of the natural homomorphism from G_k to $\mathrm{Gal}(k_{\pi,m}/k)$ and $\bar{\chi} \in \mathrm{Hom}(\mathrm{Gal}(k_{\pi,m}/k), \frac{1}{m}\mathbb{Z}/\mathbb{Z})$ with $\bar{\chi}(\tau) = 1/m$. Let $\alpha \in H^1(k, \mathrm{Pic}^0(C))[m]$ and suppose when identifying $H^1(k, \mathrm{Pic}^0(C))[m]$ with $\mathrm{Hom}(< \tau >, \mathrm{Pic}^0_k(C)[m])$ we have $\bar{S} = \alpha(\tau)$. Let $\bar{D} \in \mathrm{Pic}^0_k(C)$. We would like to show that

$$(\alpha, \bar{D}) =< \chi, F_S(D) >$$

where F_S is a function in $k(C)$ such that $(F_S) = mS$ with $S \in \bar{S}$ and $D \in \bar{D}$ such that D is prime to S.

Let $\bar{\lambda} : \frac{1}{m}\mathbb{Z}/\mathbb{Z} \to \mathbb{Z}/m\mathbb{Z}\ \bar{S}$ be the homomorphism sending $1/m$ to $\bar{S}$. Then $\varphi_\alpha = \bar{\lambda} \circ \chi$ is a function from G_k to $\mathrm{Pic}^0(C)$ that represents α.

We will relate $\delta\alpha$ in

$$H^1(G_k, \mathrm{Pic}^0(C)) \xrightarrow{\delta} H^2(G_k, \mathcal{P}(C))$$

with respect to the exact sequence

$$0 \to \mathcal{P}(C) \to \mathrm{Div}^0(C) \to \mathrm{Pic}^0(C) \to 0$$

to $\delta\chi$ in

$$H^1(G_k, \mathbb{Q}/\mathbb{Z}) \xrightarrow{\delta} H^2(G_k, \mathbb{Z})$$

with respect to the exact sequence

$$0 \to \mathbb{Z} \to \mathbb{Q} \to \mathbb{Q}/\mathbb{Z} \to 0.$$

In fact a 2-cocycle that represents $\delta\alpha$ can be directly derived and expressed in terms of F_S (see for example [7], Eq. (10.18) p. 99). However comparing $\delta\alpha$ to $\delta\chi$ will allow us to explicitly relate the pairing of $(\alpha, \bar{D})$ to the pairing $\langle \chi, F_S(D) \rangle$.

Let $\hat{\chi} \in \mathrm{Hom}(G_k, \mathbb{Q})$ be a natural lift of χ so that $\hat{\chi}$ composed with the map $\mathbb{Q} \to \mathbb{Q}/\mathbb{Z}$ is χ. Thus $\hat{\chi}$ maps to χ in $K^1(\mathbb{Q}) \to K^1(\mathbb{Q}/\mathbb{Z})$.

Let $S \in \bar{S}$ and $\lambda : \frac{1}{m}\mathbb{Z} \to \mathbb{Z}S \subset \mathrm{Div}^0_k(C) \subset \mathrm{Div}^0(C)$ be the isomorphism sending $1/m$ to S. Then $\hat{\varphi}_\alpha = \lambda \circ \hat{\chi}$ maps to φ_α in $K^1(\mathrm{Div}^0(C)) \to K^1(\mathrm{Pic}^0(C))$. (See diagram below.)

$$\begin{array}{ccccc} G_k & \to & \frac{1}{m}\mathbb{Z} & \to & \frac{1}{m}\mathbb{Z}/\mathbb{Z} \\ & & \downarrow & & \downarrow \\ & & \mathbb{Z}S & \to & \frac{\mathbb{Z}}{m\mathbb{Z}}\bar{S} \end{array}$$

By construction we have for all $\sigma, \sigma' \in G_k$,

$$d\hat{\varphi}_\alpha(\sigma, \sigma') = iS \Leftrightarrow (d\hat{\chi})(\sigma, \sigma') = \frac{i}{m},$$

hence

$$d\hat{\varphi}_\alpha(\sigma, \sigma') = m(d\hat{\chi})(\sigma, \sigma')S$$

From the general property of connecting homomorphism we know that there is a 2-cocycle $G_k \times G_k \to \mathbb{Z}$ that when composed with the inclusion map $\mathbb{Z} \to \mathbb{Q}$ is identical to $d\hat{\chi}$. Therefore it is the case that $(d\hat{\chi})(\sigma, \sigma') \in \mathbb{Z}$ for all $\sigma, \sigma' \in G_k$. Let $a_{\sigma,\sigma'} = (d\hat{\chi})(\sigma, \sigma') \in \mathbb{Z}$ for all $\sigma, \sigma' \in G_k$. Then $(a_{\sigma,\sigma'})_{\sigma,\sigma' \in G_k}$ is a 2-cocycle that represents $\delta\chi$. And we have

$$d\hat{\varphi}_\alpha(\sigma, \sigma') = m(d\hat{\chi})(\sigma, \sigma')S = ma_{\sigma,\sigma'}S = (F_S^{a_{\sigma,\sigma'}}).$$

This shows that $\delta\alpha$ can be represented by the 2-cocycle $((F_S^{a_{\sigma,\sigma'}}))_{\sigma,\sigma' \in G_k}$

So,

$$(\alpha, D) = \mathrm{inv}[(F_S(D)^{a_{\sigma,\sigma'}})_{\sigma,\sigma' \in G_k}] = \mathrm{inv}[\delta\chi \cup F_S(D)] =< \chi, F_S(D) > .$$

Theorem 2 follows.

4 The Case when k Does not Contain m-Roots of Unity

In this section we deal with the case where $\mu_m \not\subset k$.

4.1 The Multiplicative Case

To simplify the discussion, we assume throughout this subsection that m and $q-1$ are relatively prime where $q = \#\mathbb{F}$. Under the assumption, $\mathbb{F}^* = \mathbb{F}^{*m}$, hence $k^*/k^{*m} \cong \{\pi^i : i = 0, ..., m-1\}$. We also assume that m is square free, so that m and $\phi(m)$ are relatively prime.

Lemma 1. *Under the above assumptions we have an isomorphism between $H^1(G_k, \mathbb{Z}/m\mathbb{Z})$ and $Hom(Gal(k(\mu_{m^2})/k(\mu_m)), \mathbb{Z}/m\mathbb{Z})$ through which every element of $H^1(G_k, \mathbb{Z}/m\mathbb{Z})$ can be represented by an element of μ_m. More precisely ζ corresponds to $\chi \in Hom(Gal(k(\mu_{m^2})/k(\mu_m)), \mathbb{Z}/m\mathbb{Z})$ such that $\chi(\sigma) = i$ if and only if $\frac{\sigma(\zeta^{\frac{1}{m}})}{\zeta^{\frac{1}{m}}} = \zeta^i$ for all $\sigma \in Gal(k(\mu_{m^2})/k(\mu_m))$.*

To prove the lemma consider the inflation-restriction exact sequence:

$$0 \to H^1(G_k/\mathcal{I}, (\mathbb{Z}/m\mathbb{Z})^{\mathcal{I}}) \overset{\text{Inf}}{\to} H^1(G_k, \mathbb{Z}/m\mathbb{Z}) \overset{\text{Res}}{\to} H^1(\mathcal{I}, \mathbb{Z}/m\mathbb{Z})^{G_k/\mathcal{I}}.$$

Observe that $\tau^\sigma = \tau^{\chi(\sigma)}$ for $\sigma \in G_k$. Let $f \in H^1(\mathcal{I}, \mathbb{Z}/m\mathbb{Z}) = \text{Hom}(\mathcal{I}, \mathbb{Z}/m\mathbb{Z})$. Then $f(\tau^\sigma) = f(\tau^{\chi(\sigma)}) = \chi(\sigma)f(\tau)$. From this we see that f is fixed by $G_k/\mathcal{I}$ only if f is trivial. Hence $H^1(\mathcal{I}, \mathbb{Z}/m\mathbb{Z})^{G_k/\mathcal{I}} = 0$, so

$$H^1(G_k/\mathcal{I}, \mathbb{Z}/m\mathbb{Z}) = H^1(G_k/\mathcal{I}, (\mathbb{Z}/m\mathbb{Z})^{\mathcal{I}}) \overset{\text{Inf}}{\cong} H^1(G_k, \mathbb{Z}/m\mathbb{Z}).$$

Since $G_{\mathbb{F}} \cong G_k/\mathcal{I}$ and that m and $\phi(m)$ are relatively prime,

$$H^1(G_k/\mathcal{I}, \mathbb{Z}/m\mathbb{Z}) = \text{Hom}(G_k/\mathcal{I}, \mathbb{Z}/m\mathbb{Z}) \cong \text{Hom}(G_{\mathbb{F}}, \mathbb{Z}/m\mathbb{Z})$$
$$\cong \text{Hom}(\text{Gal}(\mathbb{F}(\mu_{m^2})/\mathbb{F}(\mu_m)), \mathbb{Z}/m\mathbb{Z}) \cong \text{Hom}(\text{Gal}(k(\mu_{m^2})/k(\mu_m)), \mathbb{Z}/m\mathbb{Z}).$$

The lemma follows.

From the lemma it follows that an element $\chi_\alpha \in H^1(G_k, \mathbb{Z}/m\mathbb{Z})$ is represented by an element $\alpha \in \mathbb{F}(\mu_m)^*/\mathbb{F}(\mu_m)^{*m}$. Let $L = k(\mu_m)$. Since cup product commutes with restriction and $\text{inv}_L \circ \text{Res}_{k/L} = [L : k]\ \text{inv}_k$ ([1] p. 131) it follows that

$$\langle \chi_\alpha, \pi^i \rangle_k = \text{inv}_k(\delta\chi_\alpha \cup \pi^i) = \phi(m)^{-1}\langle \chi_\alpha, \pi^i \rangle_L.$$

By Theorem 1 $\langle \chi_\alpha, \pi^i \rangle_L$ can be computed with polynomially many operations in $\mathbb{F}(\mu_m)$, together with computing a discrete logarithm in $\mathbb{F}(\mu_m)$. Therefore we have the following:

Theorem 4. *Let $q = \#\mathbb{F}$ and suppose m is square-free and relatively prime to $q-1$. Then the local duality*

$$H^1(k, \mathbb{Z}/m\mathbb{Z}) \times k^*/k^{*m} \to \mathbb{Z}/m\mathbb{Z}$$

can be computed with polynomially many operations in $\mathbb{F}(\mu_m)$, together with solving a discrete-log problem in μ_m, the subgroup of order m of $\mathbb{F}(\mu_m)^*$.

4.2 The Case of Jacobians of Curves

Let τ be a generator of $\mathrm{Gal}(k^{ur}(\pi^{\frac{1}{m}})/k^{ur})$. Let χ be the cyclotomic character so that $\sigma(\zeta) = \zeta^{\chi(\sigma)}$ for $\sigma \in G_k$. For any G_k-module B let B^χ consists of all $b \in B$ such that $\sigma b = \chi(\sigma)b$.

Lemma 2. *Let A be an abelian variety over k with good reduction at v. Suppose $\mu_m \not\subset k$ and that m is square free. Then $H^1(k, A)[m] = Hom(< \tau >, (A[m])^\chi)$.*

To prove the lemma we observe that

$$H^1(G_k, A)[m] = \mathrm{Hom}(\mathrm{Gal}(k^{ur}(\pi^{\frac{1}{m}})/k^{ur}), A[m])^{G_k/\mathcal{I}}.$$

(See for example the proof of Lemma 2.2 of [3].)

Let $\varphi \in \mathrm{Hom}(\mathrm{Gal}(k^{ur}(\pi^{\frac{1}{m}})/k^{ur}), A[m])$. Let $R_\tau = \varphi(\tau)$. Then for $t \in G_k$, we have

$$\varphi^t(\tau) = t^{-1}R_{\tau^t},$$

$$R_{\tau^t} = \varphi(\tau^t) = \varphi(\tau^{\chi(t)}) = \chi(t)R_\tau.$$

So

$$t^{-1}R_{\tau^t} = R_\tau \Leftrightarrow tR_{\tau^t} = \chi(t)R_\tau.$$

The lemma follows.

From the lemma it follows that an element $f \in H^1(k, A)[m]$ can be represented by $f(\tau) \in (A[m])^\chi \subset A(k(\mu_m))[m]$. Let A be a principally polarized abelian variety over k. For $\alpha \in H^1(k, A)[m]$ and $R \in A(k)$, the pairing between α and R is defined to be $\mathrm{inv}(\delta R \cup \beta)$ where $\beta \in H^1(k, A[m])$ is such that its image is α in $H^1(k, A[m]) \to H^1(k, A)[m]$. We have

$$(\alpha, R)_k = \mathrm{inv}_k(\delta R \cup \beta) = \phi(m)^{-1}(\alpha, R)_L.$$

Suppose A is the Jacobian variety of a curve C over a local field. Then by Theorem 2 and 3 $(\alpha, R)_L$ can be computed with polynomially many operations in $\mathbb{F}(\mu_m)$ and $\mathrm{Pic}^0_{\mathbb{F}(\mu_m)}(\tilde{C})$, together with computing a discrete logarithm in $\mathbb{F}(\mu_m)$. Therefore we have the following

Theorem 5. *Suppose $\mu_m \not\subset k$ and m is square free. Let C be a smooth projective irreducible curve over k of genus greater than 0, with a k-rational point, and good reduction at v. Then the local duality*

$$H^1(k, Pic^0(C)) \times Pic^O_k(C)/mPic^0_k(C) \to \mathbb{Z}/m\mathbb{Z}$$

is computable with polynomially many group operations in $Pic^0_{\mathbb{F}(\mu_m)}(\tilde{C})$ and field operations in $\mathbb{F}(\mu_m)$, together with solving a discrete-log problem in $\mathbb{F}(\mu_m)$.

Acknowledgement. The author would like to thank the referee for valuable comments and suggestions, and for bringing relevant results in [7] to the author's attention.

References

1. Cassels, J.W.S., Fröhlich, A.: Algebraic Number Theory. Academic Press, London (1967)
2. Frey, G., Müller, M., Rück, H.-G.: The Tate pairing and the discrete logarithm applied to elliptic curve cryptosystems. IEEE Trans. Inform. Theory 45(5), 1717–1719 (1999)
3. Frey, G., Rück, H.-G.: A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. Mathematics of Computation 62(206), 865–874 (1994)
4. Joux, A.: The Weil and Tate Pairings as Building Blocks for Public Key Cryptosystems. In: Fieker, C., Kohel, D.R. (eds.) ANTS 2002. LNCS, vol. 2369, pp. 20–32. Springer, Heidelberg (2002)
5. Lichtenbaum, S.: Duality theorems for curves over p-adic fields. Invent. Math. 7, 120–136 (1969)
6. Milne, J.S.: Arithmetic Duality Theorems. Perspectives in Mathematics, vol. 1. Academic Press, London (1986)
7. Nguyen, K.: Explicit Arithmetic of Brauer Groups – Ray Class Fields and Index Calculus, Thesis, Univesität Essen (2001)
8. Serre, J.-P.: Local Fields. Graduate Texts in Mathematics, vol. 67, Springer, Heidelberg (1979)
9. Schirokauer, O., Weber, D., Denny, T.: Discrete logarithms: The effectiveness of the index calculus method. In: Cohen, H. (ed.) ANTS 1996. LNCS, vol. 1122, pp. 337–361. Springer, Heidelberg (1996)
10. Tate, J.: WC-groups over p-adic fields. Sem. Bourbaki 156, 13 (1957)

On the Effects of Pirate Evolution on the Design of Digital Content Distribution Systems

Aggelos Kiayias*

Computer Science and Engineering, University of Connecticut
Storrs, CT, USA
aggelos@cse.uconn.edu

Abstract. A cryptographic primitive that is widely deployed commercially for digital content distribution is the subset-difference (SD) method of Naor, Naor and Lotspiech that was introduced in Crypto 2001. This encryption mechanism, called a trace and revoke scheme, is part of the Advanced Access Content System (AACS), and is used for encrypting Blu-Ray movie disks and is based on an explicit combinatorial construction of an exclusive set system. At the time of its introduction the only attacks cryptographers considered against such schemes were against the revocation and tracing algorithms. The SD method defended against them successfully and provided a superior ciphertext length compared to other known techniques : the length of the ciphertext grew only linearly with the number of revocations r; in contrast, e.g., the simpler complete subtree (CS) method requires ciphertexts of length $O(r \cdot \log N/r)$ where N is the total number of users.

In Crypto 2007 a new class of attacks was discovered against trace and revoke schemes called "pirate evolution." Pirate evolution refers to the ability of the adversary to schedule the key material it possesses in such a way so that it can withstand a great number of rounds of tracing and revocation. With the introduction of pirate evolution, the reduction of the number of rounds of pirate evolution became a design consideration for trace and revoke schemes. In 2009, Jin and Lotspiech proposed a mechanism for defending against pirate evolution in the SD method that is a tradeoff between ciphertext size and the pirate evolution bound.

In this article we provide a review of all the above results. Moreover, we compare the modified SD scheme to the CS method (similarly modified to address pirate evolution) and find that for many choices of the parameters that are relevant to practice SD can be a less preferable choice. This fact highlights the importance of considering all relevant attack scenarios when applying a specific cryptographic primitive to a certain application domain.

1 Introduction

A trace and revoke scheme is a type of encryption mechanism that is suited for digital content distribution. In particular the mechanism aims to achieve on the

* Research partly supported by NSF CAREER Award CNS-0447808.

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 223–237, 2011.
© Springer-Verlag Berlin Heidelberg 2011

fly revocation of any given set of receivers while at the same time offer a tracing capability. The tracing capability enables the revocation to occur even if no direct access to the identities of the users to be revoked is given but rather it is desired to revoke a working decoder that is given in the form of a software program accessed in black-box fashion. This type of revocation based on a decoder rather than a user identity is referred to as tracing.

Conceptually, a trace and revoke scheme was developed as a combination of two different cryptographic primitives that only partially covered the desired functionality : broadcast encryption and traitor tracing. Broadcast encryption was introduced by Fiat and Naor in [FN93] and studied further in a number of works including [GSY99, GSW00, NNL01, DF02, HS02, JHC+05, MP06]. Traitor tracing was introduced by Chor, Fiat and Naor in [CFN94] and studied further in a number of works including [SW98, NP98, KD98, BF99, SW00, SW01a, SW01b] [KY01a, KY01b, KY02, SW02, Tar03, CPP05, PST06]. The combination of the two primitives appeared first in [NP00] and explored further in [DFKY03]. Of particular interest to us now is trace and revoke schemes *for stateless receivers*, i.e., those receivers that are not required to maintain state to keep up with the broadcast transmission and thus may go arbitrarily "on and off" without loosing their signal reception capability. This type of receivers is particularly suited among other things for digital content distribution that uses disks or other physical media. Such trace and revoke schemes were proposed in [NNL01] and improved further in [HS02, JHC+05].

In [NNL01] two schemes were proposed, the complete subtree method `CS` and the subset difference `SD` method. Both of those schemes were special instances of a general framework that was using a combinatorial structure called an exclusive set system. The `SD` method was deemed especially practical given its small ciphertext length which was $2r - 1$ in the worst case where r is the number of users to be revoked, and the relatively small requirements for storage and processing for receivers. Each receiver required storage size of $O(\log^2 N)$ and required $O(\log N)$ time to recover the key required for decryption of a transmission. The perceived superiority of the `SD` method lead to its adoption as part of the advanced access content system (AACS) [AACS06] that instantiated the `SD` over the AES cipher.

In [KP07] a new class of attacks was introduced against trace and revoke schemes called pirate evolution. As pointed out in [KP07] a scheme could satisfy the security requirements of a trace and revoke scheme and still be entirely useless in practice if it is susceptible to pirate evolution with large "evolution bound." In a nutshell, a scheme is susceptible to pirate evolution if the adversary can generate a large number of decoders based on a smaller set of keys (also known as traitor keys). It was shown that in the case of `SD` the pirate evolution bound is $\Omega(t \log N)$ where t is the number of traitors. While this might seem small, it shows that a leaking incident of t traitor keys can in fact allow the production of much more decoders than corrupted users. Depending on the exact details of AACS implementation this meant that t traitor keys would yield at least $23 \cdot t$ "pirate decoders" for Blu-ray disks or even as high as $31 \cdot t$. Recognizing this

as as a critical threat, Jin and Lotspiech suggested a modification to the SD method and that was adopted in the AACS; its details were published in [JL09]. In this work it was shown how to modify the SD-method without changing the key assignment so that the number of rounds of pirate evolution can be reduced to as little as 2. This was achieved by drastically increasing the ciphertext length; in fact the length would be as high as $O(t \cdot \sqrt{N})$ as we will show here where t is the number of traitors. Given the vast storage of a Blu-Ray disk this was still deemed a small price to pay.

Nevertheless, the above casts some doubt on the optimality of the design choices for this digital content distribution mechanism. Specifically, given that the SD method was not designed to defend against pirate evolution retrofitting it to address it as suggested in [JL09] may result in sub-optimal overall performance. Indeed, as we show here this is the case for many choices of the parameters that are relevant to practice where it holds that the similarly modified CS method asymptotically matches the ciphertext length offered by the modified SD method while offering smaller key storage and smaller work required for key recovery.

Structure. In section 2 we review trace and revoke schemes, the way tracing works and how pirate evolution is defined. In section 3 we given an overview of the pirate evolution attacks against the SD method that is employed in AACS with examples. We also discuss the approach for defending against pirate evolution. Finally in section 4 we put into perspective the effects of pirate evolution in the context of digital content distribution and AACS and discuss open questions.

2 Trace and Revoke Schemes

The Subset-Cover revocation framework [NNL01] is an abstraction that can be used to formulate a variety of revocation methods. It defines a set of subsets that cover the whole user population and assigns (long-lived) keys to each subset; each user receives a collection of such keys (or derived keys). We denote by N the set of all users where $|\mathsf{N}| = N$ and $\mathsf{R} \subset \mathsf{N}$ the set of users that are to be revoked at a certain instance where $|\mathsf{R}| = r$. Note that N is not necessarily the set of currently active users but the number of all users that are anticipated in the lifetime of the system.

The goal of the sender is to transmit a message M to all users such that any $u \in \mathsf{N} \setminus \mathsf{R}$ can recover the message whereas the revoked users in R can not recover it. Note that the non-recovery property should also extend to any coalition of revoked users.

The framework is based on a collection of subsets $\{\mathsf{S}_j\}_{j \in \mathcal{J}}$ where $\mathsf{S}_j \subseteq \mathsf{N}$ such that any subset $\mathsf{S} \subseteq \mathsf{N}$ can be partitioned into disjoint subsets of $\{\mathsf{S}_j\}_{j \in \mathcal{J}}$. Each subset S_j is associated with a *long-lived* key $\mathtt{L}_j$. Users are assumed to be initialized privately with a set of keys such that u has access to $\mathtt{L}_j$ if and only if $u \in \mathsf{S}_j$. The private data assigned to user u in this intialization step will be denoted by $\mathcal{I}_u$. In particular we define $\mathcal{I}_u = \{j \in \mathcal{J} \mid u \in \mathsf{S}_j\}$ and $\mathcal{K}_u = \{\mathtt{L}_j \mid j \in \mathcal{I}_u\}$.

Given a revoked set R, the remaining users $\mathsf{N} \setminus \mathsf{R}$ are partitioned into disjoint $\{\mathsf{S}_{i_1}, \ldots, \mathsf{S}_{i_m}\} \subset \{\mathsf{S}_j\}_{j \in \mathcal{J}}$ so that $\mathsf{N} \setminus \mathsf{R} = \bigcup_{j=1}^{m} \mathsf{S}_{i_j}$.

The transmission of the message M is done in a hybrid fashion. First a random session key K is encrypted with all *long-lived* keys $\mathtt{L}_{i_j}$ corresponding to the partition, and the message M is encrypted with the session key.

Two encryption functions are being used in this framework: (1) $\mathcal{F}_K : \{0,1\}^* \mapsto \{0,1\}^*$ to encrypt the message. (2) $\mathcal{Q}_L : \{0,1\}^l \mapsto \{0,1\}^l$ to encrypt the session key.

Each broadcast ciphertext will have the following form:

$$\langle \underbrace{[i_1, i_2, \ldots i_m, \mathcal{Q}_{L_{i_1}}(K), \mathcal{Q}_{L_{i_2}}(K), \ldots \mathcal{Q}_{L_{i_m}}(K)]}_{\text{HEADER}}, \underbrace{\mathcal{F}_K(M)}_{\text{BODY}} \rangle \tag{1}$$

The receiver u decrypts a given ciphertext $\mathcal{C} = \langle [i_1, i_2, \ldots i_m, C_1, C_2, \ldots C_m], M' \rangle$ as follows: (i) Find i_j such that $u \in \mathsf{S}_{i_j}$, if not respond null, (ii) Obtain $\mathtt{L}_{i_j}$ from $\mathcal{K}_u$. (iii) Decrypt the session key: $K' = \mathcal{Q}^{-1}_{L_{i_j}}(C_j)$. (iv) Decrypt the message: $M = \mathcal{F}^{-1}_K(M')$. In [NNL01], two methods in the subset cover framework are presented called the Complete Subtree `CS` and the Subset Difference `SD`.

The Complete Subtree Method. The users are placed to the leaves of a full binary tree. In this method each node v_j of the binary tree corresponds to a subset S_j; S_j is the set of all leaves in the subtree rooted at v_j. The number of subsets in the collection is $2N - 1$ where N is the number of users (leaves). The key $\mathtt{L}_j$ for the node v_j are selected randomly and independently.

For a given set of revoked leaves R, the partition for $\mathsf{N} \setminus \mathsf{R}$ is found by computing the Steiner tree $ST(\mathsf{R})$ (the minimal subtree of the full binary tree that connects all the leaves in R). In order to compute the partition that covers the unrevoked users, suppose that $\mathsf{S}_{i_1}, \ldots, \mathsf{S}_{i_m}$ are sets of leaves belonging to subtrees of the full binary tree that "hang" off $ST(\mathsf{R})$. Note that for each tree with leaves S_{i_j}, its root v_{i_j} would be the sibling of a node in $ST(\mathsf{R})$ such that itself is not in $ST(\mathsf{R})$. The cover for $\mathsf{N} \setminus \mathsf{R}$ is $\{\mathsf{S}_{i_1}, \ldots, \mathsf{S}_{i_m}\}$. The number of subsets in any cover with N users and r revocations is at most $r \log(N/r)$. Each receiver would have to store $\log N + 1$ keys since it has $\log N + 1$ nodes along path to the root corresponding to each subset that contains the receiver. Note that the message length would be the same as the cover size, i.e., $r \log(N/r)$.

The Subset Difference Method. The subset difference method aims to decrease the cover size by increasing the number of subsets in the collection. However, increasing the number of subsets results in storing more keys on the receiver side. One can avoid excessive storage by assigning keys computationally so that the private information $\mathcal{I}_u$ contains the minimal amount necessary to compute all the necessary keys for the subsets containing u.

The users are placed to the leaves of a full binary tree. The subset $\mathsf{S}_{i,k} \in \{\mathsf{S}_j\}_{j \in \mathcal{J}}$ corresponds to a pair of nodes (v_i, v_k) where v_i is an ancestor of v_k. $\mathsf{S}_{i,k}$ is the set of all leaves in the subtree of v_i but not of v_k. The total number of subsets is $|\mathcal{J}| = O(N \log N)$. Each subset $\mathsf{S}_{i,k}$ will be assigned a key $\mathtt{L}_{i,k}$. Each

user u belongs to $O(N)$ subsets $\mathsf{S}_{i,k}$. It follows that in the information-theoretic case, each user will have to store $O(N)$ keys. To decrease the size of private information $\mathcal{I}_u$, in [NNL01], a computational method is proposed to distribute keys for each subset $\mathsf{S}_{i,k}$. It uses G, a pseudo-random sequence generator, to triple the input length. $G_L(S)$ is the left part of the output, $G_R(S)$ is the right part of the output and $G_M(S)$ is the body.

A random key $Label_i$ is assigned to each node v_i. Suppose v_j is the left (resp. right) child of v_k that is a node in the subtree rooted at v_i, then we say $Label_{i,j} = G_L(Label_{i,k})$ (resp. $Label_{i,j} = G_R(Label_{i,k})$) and $\mathtt{L}_{i,k} = G_M(Label_{i,k})$. This way, we can derive the keys $\mathtt{L}_{i,j}$ for any pair of nodes (v_i, v_j) by only applying a series of transformations to the given $Label_i$. It is also possible to derive $\mathtt{L}_{i,j}$ by applying transformations to a given $Label_{i,k}$ where both v_k and v_i are ancestors of v_j.

The private information of user u, $\mathcal{I}_u$, is defined as the set of pairs (i, k) so that $Label_{i,k} \in \mathcal{K}_u$ where v_i is the ancestor of u and v_k is hanging off the path from v_i to u, that is v_k is the sibling of an ancestor of u on the path to v_i or v_k itself is the sibling of u. Beware that user u now is a leaf in the subtree rooted at v_i but not in the subtree rooted at v_k.

Each receiver needs $O(N)$ keys, nevertheless given the computational generation of keys he has to store only $O(\log^2 N)$ keys. Processing time before decryption is at most $\log N$ applications of G. The cover size at most $2r - 1$ in the worst case scenario and $1.25r$ in the average case.

2.1 Tracing Traitors in the Subset Cover Framework

Beyond revoking sets of users that are not supposed to receive content, trace and revoke schemes are supposed to be able to disable the rogue pirate decoders which are constructed using a set of traitor's keys that are available to the pirate. One way this can be achieved is to identify a traitor given access to a pirate box and then add him to the set of revoked users.

Given that the goal of tracing is to disable the pirate box, the NNL tracing algorithm focuses on just this security goal. In the NNL setting, it is sufficient to find a "pattern" which makes the pirate box unable to decrypt.

Regarding the tracing operation, the following assumptions are used for the pirate decoder: (1) the tracing operation is black-box, i.e., it allows the tracer to examine only the outcome of the pirate decoder as an oracle. (2) the pirate decoder is not capable of recording history; (3) the pirate decoder lacks a "locking" mechanism which will prevent the tracer to pose more queries once the box detects that it is under tracing testing. (4) the pirate decoder succeeds in decoding with probability greater than or equal to a threshold q.

Based on the above, the goal of the tracing algorithm is to output *either* a non-empty subset of traitors, *or* a partition of $\mathsf{N} \setminus \mathsf{R} = \bigcup_{j=1}^{m} \mathsf{S}_{i_j}$ for the given revoked users R, such that if this partition is used to distribute content M in the framework as described above it is impossible to be decrypted by the pirate box with sufficiently high probability (larger than the threshold q); at the same time, all good users can still decrypt.

The tracing algorithm can be thought of as a repeated application of the following basic procedure that takes as input a partition: First it is tested whether the box decrypts correctly with the given partition $\bigcup_{j=1}^{m} \mathsf{S}_{i_j}$ (with probability p greater than the threshold). If not, the subset tracing outputs the partition as the output of the tracing algorithm. Otherwise, it outputs one of the subsets containing at least one of the traitors. The tracing algorithm then partitions that subset somehow and inputs the new partition (that is more "refined") to the next iteration of the basic procedure. If the subset resulting by the basic procedure contains only one possible candidate, then we can revoke that user since it is a traitor. Here is how the basic procedure works:

Let p_j be the probability that the box decodes the special tracing ciphertext

$$\langle [i_1, i_2, \ldots i_m, \mathcal{Q}_{L_{i_1}}(R), \mathcal{Q}_{L_{i_2}}(R), \ldots \mathcal{Q}_{L_{i_j}}(R), \mathcal{Q}_{L_{i_{j+1}}}(K), \ldots \mathcal{Q}_{L_{i_m}}(K)], \mathcal{F}_K(M) \rangle$$

where R is a random string of the same length as the key K. Note that $p_0 = p$ and $p_m = 0$, hence there must be some $0 < j \leq m$ for which $|p_{j-1} - p_j| \geq \frac{p}{m}$. Eventually, this leads the existence of a traitor in the subset S_{i_j} under the assumption that it is negligible to break the encryption scheme $\mathcal{Q}$ and the key assignment method.

The above can be turned into a full-fledged tracing algorithm, as long as the Subset-Cover revocation scheme satisfies the "Bifurcation property": any subset S_k can be partitioned into not extremely uneven sets S_{k_1} and S_{k_2}. Both `CS` and `SD` methods allow us to partition any subset S_k into two subsets with the Bifurcation property. For the Complete Subset, it is simply taking the subsets rooted at the children of node v_k. For the `SD` method, given $\mathsf{S}_{i,j}$ we take the subsets $\mathsf{S}_{i,c}$ and $\mathsf{S}_{c,j}$ where v_c is a child of the node v_i and v_j is on the subset rooted at v_c.

Formally, we have the following definition for tracing algorithm and encryption procedure after tracing pirate boxes to disable them recovering message:

Definition 1. *For a given set of revoked users* R *and pirate boxes* $B_1, B_2, \cdots, B_s$ *caught by the sender, the encryption function first finds a partition* $\mathcal{S}$ *which renders the pirate boxes useless and outputs the ciphertext. Let* $\mathcal{T}$ *be the tracing function outputting the partition to render the pirate boxes useless, then:* $\mathcal{T}^{B_1,B_2,\cdots,B_s}(\mathsf{R}) = \mathcal{S}$.

Denote the ciphertext created by the encryption scheme interchangeably by following notations:

$\mathcal{C} = \mathcal{E}_{\mathcal{R}}^{B_1,B_2,\cdots,B_s}(M)$ or $\mathcal{C} = \mathcal{E}_{\mathcal{S}}(M)$, *where* $\mathcal{E}_k^{-1}(\mathcal{C}) = M$ *for* $k \notin \mathsf{R}$, *and any pirate box* B_i, $0 < i \leq s$, *decrypts the ciphertext with probability less than threshold* q, *i.e.* $Prob[B_i(\mathcal{C}) = M] < q$.

According to the above definition, the sender applies tracing algorithm on the pirate boxes he has access to before broadcasting the message.

As a general model, Figure 1 represents the broadcasting procedure of the sender. [NNL01] describes how to trace traitors from many boxes as ilustrated in Figure 1.

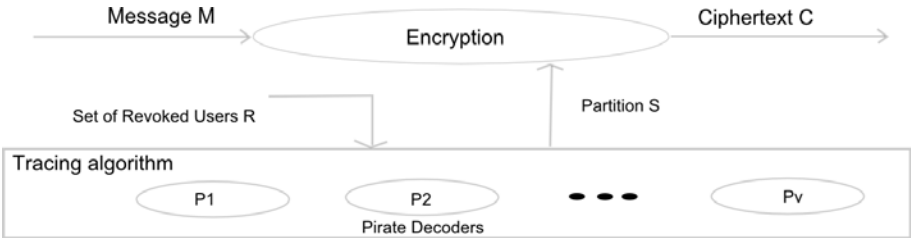


Fig. 1. Broadcasting Message in the subset cover framework, given a set of pirate decoders that must be disabled

2.2 Pirate Evolution

In this section we introduce the concept of pirate evolution. We present a game based definition that is played with the adversary which is the "evolving pirate." Let t be the number of traitor keys in the hands of the pirate. The traitor keys are made available to the pirate through a key-leaking "incident" $\mathcal{L}$ that somehow chooses a subset of size t from the set $\{\mathcal{I}_1, \ldots, \mathcal{I}_N\}$ (the set of all users' private data assigned by a function $\mathcal{G}$ with a security parameter λ). We permit $\mathcal{L}$ to be also based on the current set of revoked users R. Specifically, if $\mathsf{T} = \mathcal{L}(\mathcal{I}_1, \mathcal{I}_2, \cdots \mathcal{I}_n, t, \mathsf{R})$ then $|\mathsf{T}| = t$, $\mathsf{T} \subseteq \{\mathcal{I}_u \mid u \in \mathsf{N} \setminus \mathsf{R}\}$. This models the fact that the evolving pirate may be able to select the users that he corrupts. Separating the evolving pirate from the leaking incident is important though as it enables us to describe how a pirate can deal with leaking incidents that are not necessarily the most favorable (the pirate evolution attacks that we will describe in the sequel will operate with any given leaking incident and there will be leaking incidents that are more favorable than others). We note that partial leaking incidents can also be considered within our framework.

Once the leaking incident determines the private user data that will be available to the evolving pirate (i.e., the traitor key material), the evolving pirate $\mathcal{P}$ receives the keys and produces a "master" pirate box $\mathcal{B}$. The pirate is allowed to have oracle access to an oracle $\mathcal{E}_{\mathsf{R}}(\mathcal{M})$ that returns ciphertexts distributed according to plaintext distribution that is employed by the digital content distribution system (i.e., the access we consider is not adaptive); an adaptive version of the definition (similar to a chosen plaintext attack against symmetric encryption) is also possible.

Given the master pirate box, an iterative process is initiated: the master pirate box spawns successively a sequence of pirate decoders $B_1, B_2, \ldots$ where $B_i = \mathcal{B}(1^{t+\log N}, \ell)$ for $\ell = 1, 2, \ldots$. Note that we loosely think that the master box is simply the compact representation of a vector of pirate boxes; the time complexity allowed for its operation is polynomial in $t + \log N + \log \ell$ (this can be generalized in other contexts if needed — we found it to be sufficient for the evolving pirates strategies we present here). Each pirate box is tested whether it

decrypts correctly the plaintexts that are transmitted in the digital content distribution system with success probability at least q. The first pirate box is tested against the "initial" encryption function $\mathcal{E}_{\mathsf{R}}(\cdot)$, whereas any subsequent box is tested against $\mathcal{E}_{\mathsf{R}}^{B_1,B_2,\cdots B_{i-1}}(\cdot)$ which is the encryption that corresponds to the conjunctive revocation of the set R and the tracing of all previous pirate boxes. The iteration stops when the master pirate box $\mathcal{B}$ is incapable of producing a pirate decoder with decryption success exceeding the threshold q. Each iteration of the master box corresponds to a "generation" of pirate boxes. The number of successfully decoding pirate generations that the master box can spawn is the output of the game-based definition given below. The trace and revoke scheme is susceptible to pirate evolution if the number of generations returned by the master box is greater than t. Note that the amount of susceptibility varies with the difference between the number of generations and t; the pirate evolution bound evo is the highest number of generations any evolving pirate can produce. Formally, we have the following:

Definition 2. *Consider the game of figure 2 given two probabilistic machines $\mathcal{P}, \mathcal{L}$ and parameters $\mathsf{R} \subseteq \{1, 2, \cdots n\}$, t, $r = |\mathsf{R}|, q$. Let $PE_{\mathcal{P},\mathcal{L}}^{\mathsf{R}}(t)$ be the output of the game. We say that the trace and revoke scheme $\mathtt{TR} = (\mathcal{G}, \mathcal{Q}, \mathcal{F})$ is immune to pirate evolution with respect to key-leaking incident $\mathcal{L}$ if, for any probabilistic polynomial time adversary $\mathcal{P}$, any R and any $t \in \{1, \ldots, |\mathsf{N} - \mathsf{R}|\}$, it holds $PE_{\mathcal{P},\mathcal{L}}^{\mathsf{R}}(t) = t$. We define the pirate evolution bound $\mathsf{evo}[\mathtt{TR}]$ of a trace and revoke scheme $\mathtt{TR}$ as the supremum of all $PE_{\mathcal{P},\mathcal{L}}^{\mathsf{R}}(t)$, for any leaking incident $\mathcal{L}$, any set of revoked users R and any evolving pirate $\mathcal{P}$; note that $\mathsf{evo}[\mathtt{TR}]$ is a function of t and possibly of other parameters as well. A scheme is* susceptible *to pirate evolution if its pirate evolution bound satisfies $\mathsf{evo}[\mathtt{TR}] > t$.*

$\langle \mathcal{I}_1, \mathcal{I}_2, \cdots \mathcal{I}_N \rangle \leftarrow \mathcal{G}(1^\lambda; \rho; N)$ where $\rho \leftarrow \mathit{Coins}$
$\mathsf{T} \leftarrow \mathcal{L}(\mathcal{I}_1, \mathcal{I}_2, \cdots \mathcal{I}_n, t, \mathsf{R})$; $\mathsf{K} = \{\mathcal{K}_u \mid u : \mathcal{I}_u \in \mathsf{T}\}$
$\mathcal{B} \leftarrow \mathcal{P}^{\mathcal{E}_{\mathsf{R}}(\mathcal{M})}(\mathsf{T}, \mathsf{K})$ where $\mathcal{E}_{\mathsf{R}}(\mathcal{M})$ is an oracle that returns $\mathcal{E}_{\mathsf{R}}(m)$ with $m \leftarrow \mathcal{M}$
$\ell = 0$
repeat $\ell = \ell + 1$
  $B_\ell \leftarrow \mathcal{B}(1^{t+\log N}, \ell)$
until $\mathit{Prob}[B_\ell(\mathcal{E}_{\mathsf{R}}^{B_1,B_2,\cdots B_{\ell-1}}(m)) = m] < q$ with $m \leftarrow \mathcal{M}$
output ℓ.

Fig. 2. The attack game played with an evolving pirate

We note that immunity against pirate evolution attacks is possibly a stringent property; even though it is attainable (cf. [KP07]) it might be sacrificed in favor of efficiency.

3 Pirate Evolution and the AACS

3.1 Pirate Evolution for the Subset Difference Method

We give a brief overview of the pirate evolution attack developed in [KP07] against the Subset Difference (SD) method of [NNL01] that is part of the AACS standard [AACS06].

Recall that keys in the subset difference method are in correspondence to pairs of nodes (v_i, v_j) in a complete binary tree where the set of leaves corresponds to the set of users. We only consider pairs where v_i is an ancestor of v_j. Consider now v_c to be the child of v_i in the path from v_i to v_j. The tracing in the SD method works in such a way so that if a pirate box that uses the key (v_i, v_j) is traced then the result of tracing algorithm of [NNL01] suggests to use a revocation broadcast pattern that contains blocks encrypted with the keys $(v_i, v_c), (v_c, v_j)$.

In fact by itself, the above observation suggests a pirate evolution strategy : release gradually pirate boxes containing the keys that correspond to a forest of Steiner trees that is defined by the leaves of the complete binary tree that correspond to the traitors. Still, as shown in [KP07] one can do substantially better by taking advantage of the *frontier subsets optimization* that is performed by the tracing algorithm of [NNL01]. This is a merging operation among subsets in a a ciphertext block which substitutes sequences of keys with single keys so that the encryption functionality is preserved. Naturally, some care needs to be applied so that the optimization effect will not reverse the result of tracing. For this reason the frontier subsets optimization is only applied to sets that have not being split at an earlier stage due to tracing [NNL01] (so called "buddy subsets"). While this goes a long way in reducing the ciphertext length it opens the possibility for more extensive pirate evolution as shown in [KP07].

We will illustrate here the attack of [KP07] with an example. First, consider the state of the system as shown in figure 4. In the figure consider a setting the SD method is used for a set of 32 users and 12 of those are revoked. The corresponding ciphertext block uses 6 keys. Among the enabled users we consider 4 users that happen to be in a single subtree of size 8.

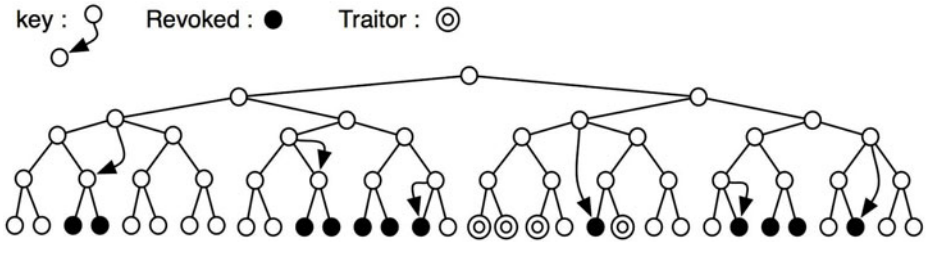


Fig. 3. An illustration of a SD ciphertext in a system with 32 users out of which 12 are revoked. The figure also shows the positions of 4 traitors among the enabled users. A mere of 6 keys is sufficient to cover all enabled users.

We demonstrate an optimal pirate evolution strategy as suggested in [KP07] that produces 11 pirate decoder generations out of the initial set of 4 traitor keys. We note that the ordering of traitors is important: a poor choice of the schedule to expend the keys may lead to a smaller number of pirate decoder generations. The optimal way to choose the traitors for any key leakage incident was shown in [KP07].

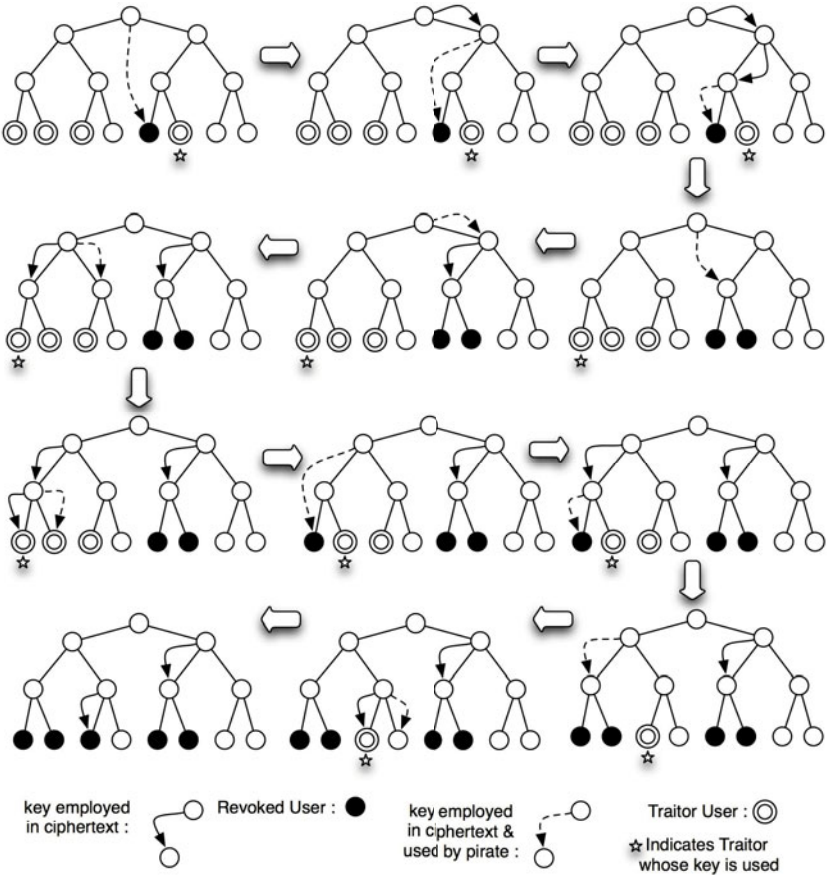

key employed in ciphertext :
Revoked User :
key employed in ciphertext & used by pirate :
Traitor User :
Indicates Traitor whose key is used

Fig. 4. Pirate evolution in action for the SD method. A set of 4 traitors produce 11 generations of pirate decoders that are successively traced until all traitors are revoked.

The "geometry" of the leaking incident is also a factor in the number of generations. The intuition here is that if the traitors are stacked closely together in the complete binary tree of all users a fewer number of pirate decoder generations can be devised.

We conclude the description with a corollary drawn from [KP07] and we refer to that work for more details on how it can be proven.

Corollary 1. *The pirate evolution bound for the* SD *method satisfies* evo[SD] $\geq t \log N$ *for* $t \leq \log N$. *It also satisfies that* evo[SD] $\geq t\frac{\log N}{2}$ *for* $t \leq \sqrt{N} \cdot \frac{\log N}{2}$.

3.2 Protecting against Pirate Evolution for the SD Method

The approach that was adopted by the AACS licensing agency to deal with the pirate evolution attack was to modify the way the tracing operation works. Specifically in [JL09] it was discussed how given a pirate decoder it is possible to split the broadcast pattern a sufficient number of times to leave little room for pirate evolution.

We illustrate this with the example in figure 5. In the figure one can see the different responses to a pirate decoder using the key illustrated on top pattern. In the case of [NNL01] the 3 traitors with the shown leaking incident may create up to 13 pirate decoder generations. In the case of [JL09], the use of the special highly fragmented broadcast pattern limits the number of pirate decoder generations to at most 6 (and possibly to as little as 2 if the fragmentation is applied recursively further down).

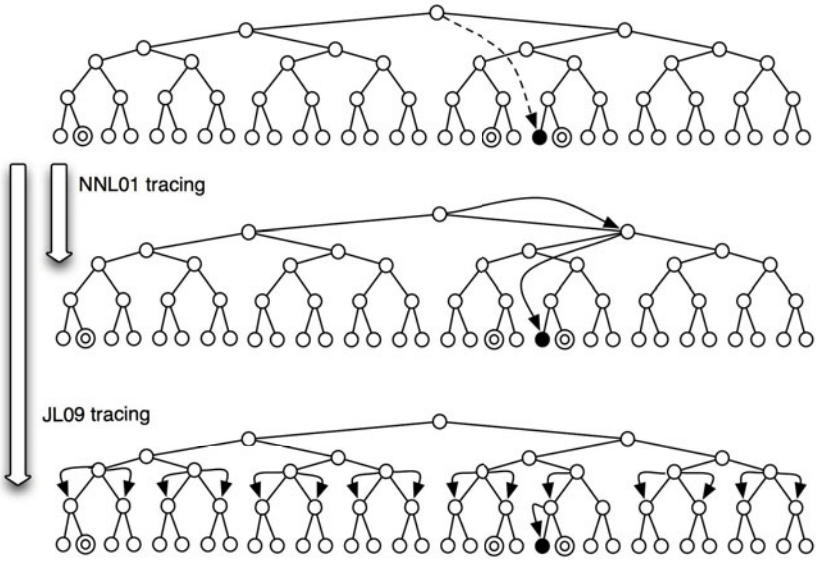


Fig. 5. Comparing the tracing mechanisms of [NNL01] and [JL09] with respect to how they respond to a pirate decoder

As is evident from the figure the defense against pirate evolution comes at a price : the number of keys needed for the broadcast pattern increases dramatically. In this example the [NNL01] broadcast pattern uses merely two keys whereas the [JL09] broadcast pattern needs 16 keys. The amount of splitting that can occur can be controlled by a parameter $b \in$. The modified subset-difference method of [JL09] fragments the broadcast pattern recursively $\log n/b$

times. Based on this the following theorem was shown in [JL09] (restated here with our notation).

Theorem 1. *The modified subset difference method* $\mathtt{SD}^b$ *for* $b \in \mathbb{N}$ *has* $\mathsf{evo}[\mathtt{SD}^b] \leq b$ *and broadcast pattern size* $O(N^{1/b})$ *after the discovery of the first pirate decoder.*

To see why the above holds, consider for simplicity that no revocation has occurred yet. In order to achieve the pirate evolution bound of b, [JL09] suggest to split the subtree containing the traitor a number of $(\log N)/b$ times. This results in a broadcast pattern of size $O(N^{1/b})$ where the users have been partitioned in subsets of size $O(N^{1-1/b})$.

We note that in the presence of t traitors the adversary can schedule the use of the keys to increase further the broadcast pattern size to $O(bt \cdot N^{1/b})$. This can be achieved by using traitor keys successively as opposed to expending all key material of a single traitor at a time. The specific choice suggested by [JL09] for AACS is $b = 2$ which results in a $O(t\sqrt{N})$ ciphertext size.

If revocation is taken into account, assuming that r users are revoked, then in the SD method there will be at most $2r-1$ keys in the broadcast pattern. Given a pirate-decoder fragmentation will occur at a single subset and will produce again $N^{1/b}$ ciphertexts (of size potentially smaller). In the worst case the behavior will not change and will result in a broadcast pattern of size $O(r+btN^{1/b})$ ciphertexts.

4 Epilogue: The Effects of Pirate Evolution

A motivation for the use of the SD method as the basis of the encryption mechanism in [AACS06] was undoubtedly its exceptionally small ciphertext length which is independent of the total number of users. This is the case as the much simpler CS method (which relates to other group key management techniques such as the logical key hierarchy [WGL98]) has smaller storage requirements at receivers : merely of $\log N$ keys and has much simpler security characteristics - in particular it does not require a key compression mechanism to reduce the required storage. In fact the number of keys needed for the SD method is $O(N)$ which is reduced to $O(\log^2 N)$ through the use of a pseudorandom number generator. This also means that the keys needed to process a transmission are not available at the receiver but must be reconstructed on the fly from the basic generator pool of size $O(\log^2 N)$. In fact this operation needs $O(\log N)$ applications of the pseudo random number generator. In contrast, in the complete subtree method there is only need to store $O(\log N)$ keys and the keys are readily available for use.

Still, the perceived advantage of the SD method was its superior ciphertext length : $2r-1$ in the worst case instead of $O(r \log N/r)$ of the complete subtree method.

Given the perceived importance of ciphertext length it seemed that SD would be the obvious choice for most implementations. This was due to the fact that

pirate evolution was not considered as an attack scenario. The only attack scenarios known and considered at the time of [NNL01] where attacks against the revocation algorithm of the underlying scheme and attacks against the tracing algorithm in the sense of evading tracing or being revoked. Taking pirate evolution into account, it was demonstrated in [KP07] that the SD method is more susceptible to it compared to the complete subtree method. Furthermore the suggested approach to deal with pirate evolution against SD that is suggested in [JL09] brings the ciphertext size to $O(bt \cdot N^{1/b})$. The approach of [JL09] can be readily applied to the complete subtree method as well. In the case of the complete subtree method in the worst case the broadcast pattern will be of size $O(r \log(N/r) + btN^{1/b})$. It follows that for any choice of $r \leq btN^{1/b}/\log(N/r)$ the two schemes become asymptotically identical in terms of ciphertext size. Given that r is not expected to be as high as $N^{1/b}$ (especially for small values of b such as $b = 2$ as suggested in [JL09]) it follows that the behavior of complete subtree and SD is asymptotically equal in terms of ciphertext overhead. On the other hand, complete subtree is superior in terms of key storage at receivers and work required to reconstruct the key used in a transmission. This outcome highlights the importance of having the right security considerations when choosing and optimizing a certain security system for a target application domain.

Given the above, an interesting open question is the design of an optimal scheme for a given target pirate evolution bound b (or showing that one of the existing schemes is optimal in some sense). Note that due to the variety of parameters (storage, ciphertext overhead, key recovery) there can be schemes offering various tradeoffs between them for a fixed target evolution bound.

References

[AACS06] AACS Specifications (2006), http://www.aacsla.com/specifications/

[BF99] Boneh, D., Franklin, M.K.: An Efficient Public Key Traitor Scheme. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 338–353. Springer, Heidelberg (1999)

[BSW06] Boneh, D., Sahai, A., Waters, B.: Fully Collusion Resistant Traitor Tracing with Short Ciphertexts and Private Keys. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 573–592. Springer, Heidelberg (2006)

[BS98] Boneh, D., Shaw, J.: Collusion-Secure Fingerprinting for Digital Data. IEEE Transactions on Information Theory 44(5), 1897–1905 (1998)

[CPP05] Chabanne, H., Phan, D.H., Pointcheval, D.: Public traceability in traitor tracing schemes. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 542–558. Springer, Heidelberg (2005)

[CFN94] Chor, B., Fiat, A., Naor, M.: Tracing Traitors. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 257–270. Springer, Heidelberg (1994)

[CFNP00] Chor, B., Fiat, A., Naor, M., Pinkas, B.: Tracing Traitors. IEEE Transactions on Information Theory 46(3), 893–910 (2000)

[DF02] Dodis, Y., Fazio, N.: Public Key Broadcast Encryption for Stateless Receivers. In: Feigenbaum, J. (ed.) DRM 2002. LNCS, vol. 2696, pp. 61–80. Springer, Heidelberg (2003)

[DFKY03] Dodis, Y., Fazio, N., Kiayias, A., Yung, M.: Scalable public-key tracing and revoking, PODC 2003. In: Proceedings of the Twenty-Second ACM Symposium on Principles of Distributed Computing (PODC 2003), Boston, Massachusetts, July 13-16, pp. 190–199 (2003)

[FN93] Fiat, A., Naor, M.: Broadcast Encryption. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 480–491. Springer, Heidelberg (1994)

[FT01] Fiat, A., Tassa, T.: Dynamic Traitor Tracing. Journal of Cryptology 4(3), 211–223 (2001)

[GGM86] Goldreich, O., Goldwasser, S., Micali, S.: How to Construct Random Functions. J. of the ACM 33(4), 792–807 (1986)

[GSY99] Gafni, E., Staddon, J., Yin, Y.L.: Efficient Methods for Integrating Traceability and Broadcast Encryption. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 372–387. Springer, Heidelberg (1999)

[GSW00] Garay, J.A., Staddon, J., Wool, A.: Long-Lived Broadcast Encryption. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 333–352. Springer, Heidelberg (2000)

[HS02] Halevy, D., Shamir, A.: The LSD Broadcast Encryption Scheme. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 47–60. Springer, Heidelberg (2002)

[JHC+05] Jho, N.-S., Hwang, J.Y., Cheon, J.H., Kim, M.-H., Lee, D.-H., Yoo, E.S.: One-Way Chain Based Broadcast Encryption Schemes. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 559–574. Springer, Heidelberg (2005)

[JL09] Jin, H., Lotspiech, J.: Defending against the Pirate Evolution Attack. In: Bao, F., Li, H., Wang, G. (eds.) ISPEC 2009. LNCS, vol. 5451, pp. 147–158. Springer, Heidelberg (2009)

[KP07] Kiayias, A., Pehlivanoglu, S.: Pirate evolution: How to make the most of your traitor keys. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 448–465. Springer, Heidelberg (2007)

[KY01a] Kiayias, A., Yung, M.: Self Protecting Pirates and Black-Box Traitor Tracing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 63–79. Springer, Heidelberg (2001)

[KY01b] Kiayias, A., Yung, M.: On Crafty Pirates and Foxy Tracers. In: Sander, T. (ed.) DRM 2001. LNCS, vol. 2320, pp. 22–39. Springer, Heidelberg (2002)

[KY02] Kiayias, A., Yung, M.: Traitor Tracing with Constant Transmission Rate. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 450–465. Springer, Heidelberg (2002)

[KD98] Kurosawa, K., Desmedt, Y.: Optimum Traitor Tracing and Asymmetric Schemes. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 145–157. Springer, Heidelberg (1998)

[MP06] Micciancio, D., Panjwani, S.: Corrupting One vs. Corrupting Many: The Case of Broadcast and Multicast Encryption. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 70–82. Springer, Heidelberg (2006)

[NNL01] Naor, D., Naor, M., Lotspiech, J.: Revocation and Tracing Schemes for Stateless Receivers. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 41–62. Springer, Heidelberg (2001)

[NP98] Naor, M., Pinkas, B.: Threshold Traitor Tracing. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 502–517. Springer, Heidelberg (1998)

[NP00] Naor, M., Pinkas, B.: Efficient Trace and Revoke Schemes. In: Frankel, Y. (ed.) FC 2000. LNCS, vol. 1962, pp. 1–20. Springer, Heidelberg (2001)

[NR97] Naor, M., Reingold, O.: Number-Theoretic Constructions of Efficient Pseudo-Random Functions. In: 38th Annual Symposium on Foundations of Computer Science, FOCS 1997, Miami Beach, Florida, USA, October 19-22, pp. 458–467. IEEE Computer Society, Los Alamitos (1997)

[Pfi96] Pfitzmann, B.: Trials of Traced Traitors. In: Anderson, R.J. (ed.) IH 1996. LNCS, vol. 1174, pp. 49–63. Springer, Heidelberg (1996)

[PST06] Phan, D.H., Safavi-Naini, R., Tonien, D.: Generic Construction of Hybrid Public Key Traitor Tracing with Full-Public-Traceability, pp. 264–275

[SW00] Safavi-Naini, R., Wang, Y.: Sequential Traitor Tracing. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 316–332. Springer, Heidelberg (2000)

[SW01a] Safavi-Naini, R., Wang, Y.: Collusion Secure q-ary Fingerprinting for Perceptual Content. In: Sander, T. (ed.) DRM 2001. LNCS, vol. 2320, pp. 57–75. Springer, Heidelberg (2002)

[SW01b] Safavi-Naini, R., Wang, Y.: New Results on Frameproof Codes and Traceability Schemes. IEEE Transactions on Information Theory 47(7), 3029–3033 (2001)

[SW02] Safavi-Naini, R., Wang, Y.: Traitor Tracing for Shortened and Corrupted Fingerprints. In: Feigenbaum, J. (ed.) DRM 2002. LNCS, vol. 2696, pp. 81–100. Springer, Heidelberg (2003)

[SSW00] Jessica, N., Staddon, D.R.: Combinatorial Properties of Frameproof and Traceability Codes. IEEE Transactions on Information Theory 47(3), 1042–1049 (2001)

[SW98] Stinson, D.R., Wei, R.: Combinatorial Properties and Constructions of Traceability Schemes and Frameproof Codes. SIAM Journal on Discrete Math. 11(1), 41–53 (1998)

[WNR04] Wang, P., Ning, P., Reeves, D.S.: Storage-efficient stateless group key revocation. In: Zhang, K., Zheng, Y. (eds.) ISC 2004. LNCS, vol. 3225, pp. 25–38. Springer, Heidelberg (2004)

[WGL98] Wong, C.K., Gouda, M., Lam, S.: Secure Group Communications Using Key Graphs. In: SIGCOMM (1998)

[Tar03] Tardos, G.: Optimal probabilistic fingerprint codes. In: Proceedings of the 35th Annual ACM Symposium on Theory of Computing, San Diego, CA, USA, June 9-11, pp. 116–125. ACM, New York (2003)

Arithmetic of Split Kummer Surfaces: Montgomery Endomorphism of Edwards Products

David Kohel

Institut de Mathématiques de Luminy
Université de la Méditerranée
163, avenue de Luminy, Case 907
13288 Marseille Cedex 9
France

Abstract. Let E be an elliptic curve, $\mathscr{K}_1$ its Kummer curve $E/\{\pm 1\}$, E^2 its square product, and $\mathscr{K}_2$ the split Kummer surface $E^2/\{\pm 1\}$. The addition law on E^2 gives a large endomorphism ring, which induce endomorphisms of $\mathscr{K}_2$. With a view to the practical applications to scalar multiplication on $\mathscr{K}_1$, we study the explicit arithmetic of $\mathscr{K}_2$.

1 Introduction

Let A be an abelian group, whose group law is expressed additively. Let $\mathrm{M}_2(\mathbb{Z})$ be the subring of $\mathrm{End}(A^2)$, acting as

$$\alpha(x, y) = (ax + by, cx + dy) \text{ where } \alpha = \begin{pmatrix} a & b \\ c & d \end{pmatrix}.$$

Define endomorphisms σ and φ_i by

$$\sigma = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \ \varphi = \varphi_0 = \begin{pmatrix} 2 & 0 \\ 1 & 1 \end{pmatrix}, \text{ and } \varphi_1 = \sigma\varphi\sigma = \begin{pmatrix} 1 & 1 \\ 0 & 2 \end{pmatrix}.$$

The Montgomery ladder for scalar multiplication by an integer n is expressed on A^2 by the recursion

$$v_r = (0, x) \text{ and } v_i = \varphi_{n_i}(v_{i+1}) \text{ for } i = r - 1, \ldots, 1, 0,$$

where n has binary representation $n_{r-1} \ldots n_1 n_0$. The successive steps v_i in the ladder are of the form $(mx, (m + 1)x)$ and $v_0 = (nx, (n + 1)x)$, from which we output nx (see Montgomery [10] and Joye [8] for general formulation). We refer to φ as the *Montgomery endomorphism*.

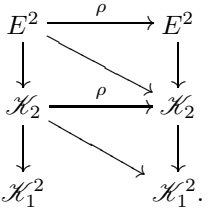
Since -1 is an automorphism in the center of $\mathrm{M}_2(\mathbb{Z})$, an endomorphism of A^2 also acts on the quotient $A^2/\{\pm 1\}$. In particular, we will derive expressions of the above operators on the split Kummer surface $\mathscr{K}_2 = E^2/\{\pm 1\}$ associated to an elliptic curve E.

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 238–245, 2011.
© Springer-Verlag Berlin Heidelberg 2011

Prior work has focused on Kummer curves $\mathscr{K}_1 = E/\{\pm 1\} \cong \mathbb{P}^1$, determined by the quotient $\pi : E \to \mathscr{K}_1$, often expressed as operating only on the x-coordinate of a Weierstrass model (see Montgomery [10], Brier and Joye [3] and Izu and Takagi [7]). Such methods consider the full quotient $\mathscr{K}_1^2 = E^2/\{(\pm 1, \pm 1)\}$. For this approach one takes the endomorphism

$$\rho = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix},$$

arising in duplication formulas for theta functions [11]. This endomorphism satisfies $\rho^2 = 2$, giving a factorization of 2 in $\mathrm{End}(E^2)$, and induces an endomorphism of $\mathscr{K}_2$, which we also refer to as ρ. This gives a commutative diagram:



Although ρ does not extend to an endomorphism of $\mathscr{K}_1^2$ we obtain a system of polynomial equations in $\mathscr{K}_1^2 \times \mathscr{K}_1^2$ from the graph:

$$\Gamma_\rho = \big\{\big((P,Q), \rho(P,Q)\big) : (P,Q) \in E^2\big\} \subset E^2 \times E^2.$$

One recovers $\pi(P+Q)$ from specializing this sytem at known points $\pi(P)$, $\pi(Q)$ and $\pi(P-Q)$. By considering the partial quotient $\mathscr{K}_2$ as a double cover of $\mathscr{K}_1^2$, we obtain endmorphisms of $\mathscr{K}_2$ induced by the isogenies ρ as well as φ_0 and φ_1.

Since the structure of addition laws of abelian varieties, or isogenies in general, depends intrinsically on the embedding in projective space (see [6], [9]), we develop specific models for the Kummer surface $\mathscr{K}_2$ associated to a model of an elliptic curve E with prescribed embedding. For this purpose we investigate Edwards models for elliptic curves embedded in $\mathbb{P}^3$.

2 Projective Embeddings of a Kummer Variety $\mathscr{K}$

Let k be a field of characteristic different from 2 and A/k an abelian variety. An addition law on A is defined by Lange and Ruppert [9] to be a polynomial representative for the addition morphism $A^2 \to A$. Such maps depend in an essential way on its projective embedding. Similarly, the explicit polynomial maps for morphisms of the Kummer variety $\mathscr{K} = A/\{\pm 1\}$ depend on a choice of its projective embedding. We approach the problem of embedding $\mathscr{K}$ in the following way.

Let $i : A \to \mathbb{P}^r$ be a projectively normal embedding (see [6] for a definition and motivation for this hypothesis), determined by a symmetric invertible sheaf $\mathscr{L} = \mathcal{O}_A(1) = i^*\mathcal{O}_{\mathbb{P}^r}(1)$ and let $\pi : A \to \mathscr{K}$ be the projection morphism.

We say that an embedding $j : \mathscr{K} \to \mathbb{P}^s$ is *compatible* with $i : A \to \mathbb{P}^r$ if π is represented by a linear polynomial map. In terms of the invertible sheaf $\mathscr{L}_1 = \mathcal{O}_{\mathscr{K}}(1) = j^*\mathcal{O}_{\mathbb{P}^s}(1)$, this condition is equivalent to:

$$\mathrm{Hom}(\pi^*\mathscr{L}_1, \mathscr{L}) \cong \Gamma(A, \pi^*\mathscr{L}_1^{-1} \otimes \mathscr{L}) \neq 0,$$

where $\Gamma(A, \mathscr{M})$ is the space of global sections for a sheaf $\mathscr{M}$. If we have $\pi^*\mathscr{L}_1 \cong \mathscr{L}$ then $\mathrm{Hom}(\pi^*\mathscr{L}_1, \mathscr{L}) \cong k$, and π admits a unique linear polynomial map, up to scalar.

Conversely we can construct an embedding of $\mathscr{K}$ comptable with given $i : A \to \mathbb{P}^r$ as follows. The condition that $i : A \to \mathbb{P}^r$ is projectively normal is equivalent to an isomorphism of graded rings

$$k[X_0, X_1, \ldots, X_r]/I_A = \bigoplus_{n=0}^{\infty} \Gamma(A, \mathscr{L}^n),$$

where I_A is the defining ideal for A in $\mathbb{P}^r$. We fix an isomorphism $\mathscr{L} \cong [-1]^*\mathscr{L}$, from which we obtain an eigenspace decomposition of the spaces $\Gamma(A, \mathscr{L}^n)$:

$$\Gamma(A, \mathscr{L}^n) = \Gamma(A, \mathscr{L}^n)^+ \oplus \Gamma(A, \mathscr{L}^n)^-.$$

The sign is noncanonical, but we may choose the sign for the isomorphism $\mathscr{L} \cong [-1]^*\mathscr{L}$ such that $\dim \Gamma(A, \mathscr{L})^+ \geq \dim \Gamma(A, \mathscr{L})^-$. Setting $V = \Gamma(A, \mathscr{L})^+$, we define $j : \mathscr{K} \to \mathbb{P}^s$ by the image of A in $\mathbb{P}^s = \mathbb{P}(V)$. This defines the sheaf $\mathscr{L}_1 = j^*\mathcal{O}_{\mathbb{P}^s}(1)$ and gives a homomorphism $\pi^*\mathscr{L}_1 \to \mathscr{L}$.

In what follows we carry out this construction to determine projective embeddings for the Kummer varieties $\mathscr{K}_1$ and $\mathscr{K}_2$ associated to an elliptic curve embedded as an Edwards model in $\mathbb{P}^3$, and study the form of the endomorphisms σ, φ and ρ.

3 Edwards Model and Projective Embeddings of $\mathscr{K}_1$

Let E be an elliptic curve embedded in $\mathbb{P}^3$ as an Edwards model (see Edwards [4], Bernstein and Lange [1], and Hisil et al. [5] or Kohel [6] for this form):

$$X_0^2 + dX_3^2 = X_1^2 + X_2^2, \quad X_0X_3 = X_1X_2,$$

with identity $\mathrm{O} = (1 : 0 : 1 : 0)$, and negation map

$$[-1](X_0 : X_1 : X_2 : X_3) = (X_0 : -X_1 : X_2 : -X_3).$$

The eigenspace decomposition for $\Gamma(E, \mathscr{L})$ is

$$\Gamma(E, \mathscr{L}) = \bigoplus_{i=1}^{4} kX_i = (kX_0 \oplus kX_1) \oplus (kX_2 \oplus kX_3).$$

The Kummer curve of E is $\mathscr{K}_1 \cong \mathbb{P}^1$, with quotient map

$$(X_0 : X_1 : X_2 : X_3) \mapsto (X_0 : X_2) = (X_1 : X_3).$$

We can now express the scalar multiplication by 2 on $\mathscr{K}_1$ in terms of coordinate functions X_0, X_1 on $\mathscr{K}_1$.

Lemma 1. *The duplication morphism* $[2] : \mathscr{K}_1 \to \mathscr{K}_1$ *is uniquely represented by the polynomial map*

$$(X_0 : X_1) \mapsto ((d-1)X_0^4 - d(X_0^2 - X_1^2)^2 : (X_0^2 - X_1^2)^2 + (d-1)X_1^4).$$

Proof. The correctness of the polynomial map can be directly verified by the fact that the known endomorphisms [2] on E commutes with π and the above polynomial map for [2] on $\mathscr{K}_1$. The uniqueness follows from the existence of the above degree four polynomial expressions, since from $\deg([2]) = 4$ we obtain $[2]^*\mathscr{L}_1 \cong \mathscr{L}_1^4$. Since degree n polynomial expressions for a morphism ψ are in bijection with

$$\mathrm{Hom}(\psi^*\mathscr{L}_1, \mathscr{L}_1^n) \cong \Gamma(E, \psi^*\mathscr{L}_1^{-1} \otimes \mathscr{L}_1^n),$$

the result follows. □

4 Segre Embedings and Projective Products

In general a projective model behaves well with respect to the theory. In order to characterize a product $X \times Y$ with $X \subseteq \mathbb{P}^r$ and $Y \subseteq \mathbb{P}^s$ we apply the Segre embedding $S : \mathbb{P}^r \times \mathbb{P}^s \to \mathbb{P}^{rs+r+s}$ given by

$$\big((X_0 : X_1 : \cdots : X_r), (Y_0 : Y_1 : \cdots : Y_s)\big) \longmapsto (X_0Y_0 : X_1Y_0 : \cdots : X_rY_s),$$

and consider the image $S(X \times Y)$ in $\mathbb{P}^{rs+r+s}$.

For $r = s = 1$, we have $(r+1)+(s+1) = 4$ coordinates to represent a point in $\mathbb{P}^1 \times \mathbb{P}^1$ and $(r+1)(s+1) = 4$ coordinates for a point in $\mathbb{P}^3$. For higher degrees or powers $\mathbb{P}^{r_1} \times \cdots \times \mathbb{P}^{r_t}$ the Segre embedding becomes unwieldy for explicit computation.

In particular, for the product $\mathscr{K}_1^2 \cong \mathbb{P}^1 \times \mathbb{P}^1$ this gives the embedding of $\mathscr{K}_1^2$ in $\mathbb{P}^3$ as the hypersurface $U_0U_3 = U_1U_2$, given by

$$\big((X_0 : X_1), (Y_0 : Y_1)\big) \longmapsto (U_0 : U_1 : U_2 : U_3) = (X_0Y_0 : X_1Y_0 : X_0Y_1 : X_1Y_1).$$

The inverse is given by the product of projections $\pi_1 : S(\mathscr{K}_1^2) \to \mathscr{K}_1$

$$(U_0 : U_1 : U_2 : U_3) \longmapsto (U_0 : U_1) = (U_2 : U_3),$$

and $\pi_2 : S(\mathscr{K}_1^2) \to \mathscr{K}_1$

$$(U_0 : U_1 : U_2 : U_3) \longmapsto (U_0 : U_2) = (U_1 : U_3).$$

Each projection is represented locally by a two-dimensional space of linear polynomial maps, but no such map defines π_i globally as a morphism.

We use the Segre embedding $\mathscr{K}_1^2 \to S(\mathscr{K}_1^2)$ to provide a projective embedding for $\mathscr{K}_1^2$ and construct $\mathscr{K}_2$ as a double cover of $S(\mathscr{K}_1^2)$ in $S(\mathscr{K}_1^2) \times \mathbb{P}^1 \subseteq \mathbb{P}^3 \times \mathbb{P}^1$. To preserve the compactness of the representation we work with the model in $\mathbb{P}^3 \times \mathbb{P}^1$, rather than its model in $\mathbb{P}^7$, however we give this model in Theorem 1.

In order to define a morphism $\mathscr{K}_2 \to \mathscr{K}_2$ it suffices to make use of the factorization through $\mathscr{K}_1^2 \times \mathbb{P}^1$ to each of the products. Thus a morphism $\psi : X \to \mathscr{K}_2$ is determined by three maps $\psi_i = \pi_i \circ \psi$ for $1 \leq i \leq 3$, and a composition with a Segre embedding of $\mathscr{K}_1^2$ to $\mathbb{P}^3$ gives the map to $\mathscr{K}_2$ in $\mathbb{P}^3 \times \mathbb{P}^1$. We note, however, that expansion of polynomial maps for this factorization $S \circ (\pi_1 \times \pi_2)$ may yield polynomial maps of higher degree than $\mathscr{K}_2 \to S(\mathscr{K}_1^2)$ directly (see Theorem 1).

Note. Despite the isomorphism $\mathscr{K}_1 \cong \mathbb{P}^1$, and even equality under the projective embedding, we write $\mathscr{K}_1^2$ and $\mathscr{K}_1^2 \times \mathbb{P}^1$ rather than $(\mathbb{P}^1)^2$ and $(\mathbb{P}^1)^3$ in order to reflect the distinguished role of the two Kummer curves in this product.

5 Edwards Model and Projective Embeddings of $\mathscr{K}_2$

We now describe the embeddings of $\mathscr{K}_2$ as a double cover of $\mathscr{K}_1^2$.

Theorem 1. *Let $E : X_0^2 + dX_3^2 = X_1^2 + X_2^2, X_0X_3 = X_1X_3$ be an elliptic curve in $\mathbb{P}^3$ with identity $\mathrm{O} = (1 : 0 : 1 : 0)$. The Kummer surface $\mathscr{K}_2$ has a model as a hypersurface in $\mathscr{K}_1^2 \times \mathbb{P}^1$ given by*

$$(X_0^2 - X_1^2)(Y_0^2 - Y_1^2)Z_0^2 = (X_0^2 - dX_1^2)(Y_0^2 - dY_1^2)Z_1^2,$$

with base point $\pi(\mathrm{O}) = ((1 : 1), (1 : 1), (1 : 0))$, and projection $E^2 \to \mathscr{K}_2$ given by $\pi_1(P, Q) = (X_0 : X_2)$, $\pi_2(P, Q) = (Y_0 : Y_2)$, and

$$\pi_3(P, Q) = (X_0Y_0 : X_1Y_1) = (X_2Y_0 : X_3Y_1) = (X_0Y_2 : X_1Y_3) = (X_2Y_2 : X_3Y_3),$$

where $(P, Q) = \big((X_0 : X_1 : X_2 : X_3), (Y_0 : Y_1 : Y_2 : Y_3)\big)$.

Under the Segre embedding $S : \mathscr{K}_1^2 \mapsto \mathbb{P}^3$, this determines the variety in $\mathbb{P}^3 \times \mathbb{P}^1$ cut out by

$$(U_0^2 - U_1^2 - U_2^2 + U_3^2)Z_0^2 = (U_0^2 - dU_1^2 - dU_2^2 + d^2U_3^2)Z_1^2,$$

on the hypersurface $U_0U_3 = U_1U_2$ defining $S(\mathscr{K}_1^2)$. The Segre embedding of $\mathscr{K}_2$ in $\mathbb{P}^7$ is cut out by the quadratic relation

$$T_0^2 - T_1^2 - T_2^2 + T_3^2 = T_4^2 - dT_5^2 - dT_6^2 + d^2T_7^2,$$

on the image of the Segre embedding of $(\mathbb{P}^1)^3 \to \mathbb{P}^7$, determined by:

$$\begin{gathered} T_0T_3 = T_1T_2,\ T_0T_5 = T_1T_4,\ T_0T_6 = T_2T_4, \\ T_0T_7 = T_3T_4,\ T_1T_6 = T_3T_4,\ T_1T_7 = T_3T_5, \\ T_2T_5 = T_3T_4,\ T_2T_7 = T_3T_6,\ T_4T_7 = T_5T_6. \end{gathered}$$

The morphism to $E^2 \to S(\mathscr{K}_2) \subseteq \mathbb{P}^7$ is determined by:

$$(X_0Y_0 : X_2Y_0 : X_0Y_2 : X_2Y_2, X_1Y_1 : X_3Y_1 : X_1Y_3 : X_3Y_3).$$

Proof. The quadratic relation for $\mathscr{K}_2$ in $\mathscr{K}_1^2 \times \mathbb{P}^1$:

$$(X_0^2 - X_1^2)(Y_0^2 - Y_1^2)Z_0^2 = (X_0^2 - dX_1^2)(Y_0^2 - dY_1^2)Z_1^2,$$

follows by pulling back the relation to E^2 by

$$\pi^*(Y_1/Y_0) = (Y_2/Y_0),\ \pi^*(X_1/X_0) = (X_2/X_0),\ \pi^*(Z_1/Z_0)^2 = (X_1Y_1/X_0Y_0)^2.$$

Since the morphism maps through $E^2/\{\pm 1\}$, defines a double cover of $\mathscr{K}_1^2$, and is irreducible, we conclude that the quadratic relation determines $\mathscr{K}_2$. The remaining models follow by tracing this quadratic relation through the Segre embeddings.

The last model, in $\mathbb{P}^7$, can be interpreted as coming from the construction of Section 2, applied to the Segre embedding of E^2 in $\mathbb{P}^{15}$. The sixteen-dimensional space of global sections splits into two eight-dimensional subspaces, for which

$$\{X_0Y_0 : X_2Y_0 : X_0Y_2 : X_2Y_2, X_1Y_1 : X_3Y_1 : X_1Y_3 : X_3Y_3\}$$

forms a basis for the plus one eigenspace. The compatibility of the maps from E^2 is verified by projecting from the models in $\mathbb{P}^7$ and $\mathbb{P}^3 \times \mathbb{P}^1$ to $\mathscr{K}_1^2 \times \mathbb{P}^1$. □

The description of the maps in the previous theorem, together with the action of $[-1]$ on the Edwards model, implies the next corollary.

Corollary 1. *The automorphism $\sigma : E^2 \to E^2$ given by $(P, Q) \mapsto (Q, P)$ induces the automorphism of $\mathscr{K}_2$ in the respective models in $\mathscr{K}_1^2 \times \mathbb{P}^1$, $\mathbb{P}^3 \times \mathbb{P}^1$ and $\mathbb{P}^7$:*

$$\big((X_0 : X_1), (Y_0 : Y_1), (Z_0 : Z_1)\big) \mapsto \big((Y_0 : Y_1), (X_0 : X_1), (Z_0 : Z_1)\big),$$

$$\big((U_0 : U_1 : U_2 : U_3), (Z_0 : Z_1)\big) \mapsto \big((U_0 : U_2 : U_1 : U_3), (Z_0 : Z_1)\big),$$

$$(T_0 : T_1 : T_2 : T_3 : T_4 : T_5 : T_6 : T_7) \mapsto (T_0 : T_2 : T_1 : T_3 : T_4 : T_6 : T_5 : T_7).$$

The automorphism $\iota : \mathscr{K}_2 \to \mathscr{K}_2$ induced by the automorphisms $[-1] \times [1]$ and $[1] \times [-1]$ of E^2 is given by:

$$\big((X_0 : X_1), (Y_0 : Y_1), (Z_0 : Z_1)\big) \mapsto \big((X_0 : X_1), (Y_0 : Y_1), (Z_0 : -Z_1)\big),$$

$$\big((U_0 : U_1 : U_2 : U_3), (Z_0 : Z_1)\big) \mapsto \big((U_0 : U_1 : U_2 : U_3), (Z_0 : -Z_1)\big),$$

$$(T_0 : T_1 : T_2 : T_3 : T_4 : T_5 : T_6 : T_7) \mapsto (T_0 : T_1 : T_2 : T_3 : -T_4 : -T_5 : -T_6 : -T_7).$$

6 Endomorphisms of Kummer Surfaces $\mathscr{K}_2$

We are now able to define polynomial maps for the Montgomery endomorphism φ, where ρ, τ, and φ are the endomorphisms

$$\varphi = \begin{pmatrix} 2 & 0 \\ 1 & 1 \end{pmatrix} \text{ and } \rho = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \text{ and } \tau = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix},$$

as elements of $\mathrm{M}_2(\mathbb{Z})/\{\pm 1\}$. In addition we recall the definitions

$$\iota = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \text{ and } \sigma = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},$$

and note the commuting relations $\rho \circ \iota = \sigma \circ \rho$ and $\rho \circ \sigma = \iota \circ \rho$ for ι, σ, and ρ.

Explicit polynomial maps for the Montgomery endomorphism φ on $\mathscr{K}_2$ follow from the identities

$$\varphi_0 = \varphi = \tau \circ \sigma\rho \text{ and } \varphi_1 = \sigma \circ \varphi \circ \sigma.$$

As a consequence the Montgomery ladder can be expressed in terms of the automorphisms σ, ι, and endomorphisms ρ and τ. The following two theorems, whose proof follows from standard addition laws on the Edwards model (see Bernstein and Lange [1], [2], Hisil [5], and Kohel [6]), and verification of the commutativity relations $\pi \circ \psi = \psi \circ \pi$ for an endomorphism ψ.

Theorem 2. *The projections of the endomorphisms $\rho : \mathscr{K}_2 \to \mathscr{K}_2$ are uniquely represented by polynomials of bidegree $(1,1)$, $(1,1)$, and $(2,0)$, explicitly:*

$$\begin{aligned}
\pi_1 \circ \tau\big((U_0 : U_1 : U_2 : U_3), (Z_0 : Z_1)\big) &= (U_0Z_0 - dU_3Z_1 : -U_0Z_1 + U_3Z_0) \\
\pi_2 \circ \tau\big((U_0 : U_1 : U_2 : U_3), (Z_0 : Z_1)\big) &= (U_0Z_0 + dU_3Z_1 : U_0Z_1 + U_3Z_0), \\
\pi_3 \circ \tau\big((U_0 : U_1 : U_2 : U_3), (Z_0 : Z_1)\big) &= (U_0^2 - dU_3^2 \ : \ -U_1^2 + U_2^2).
\end{aligned}$$

The projection $\rho : \mathscr{K}_2 \to S(\mathscr{K}_1{}^2)$ admits a two-dimensional space of polynomial maps of bidegree $(2,1)$ spanned by:

$$\begin{aligned}
&\big(U_0^2 - dU_1^2 - dU_2^2 + dU_3^2)Z_0 \ : \\
&\quad -(d-1)U_0U_3Z_0 - (U_0^2 - dU_1^2 - dU_2^2 + d^2U_3^2)Z_1 \ : \\
&\quad -(d-1)U_0U_3Z_0 + (U_0^2 - dU_1^2 - dU_2^2 + d^2U_3^2)Z_1 \ : \\
&\quad -(U_0^2 - U_1^2 - U_2^2 + dU_3^2)Z_0 \ \big) \\
&\big((U_0^2 - dU_1^2 - dU_2^2 + dU_3^2)Z_1 \ : \\
&\quad -(d-1)U_0U_3Z_1 - (U_0^2 - U_1^2 - U_2^2 + U_3^2)Z_0 \ : \\
&\quad -(d-1)U_0U_3Z_1 + (U_0^2 - U_1^2 - U_2^2 + U_3^2)Z_0 \ : \\
&\quad -(U_0^2 - U_1^2 - U_2^2 + dU_3^2)Z_1\big).
\end{aligned}$$

Theorem 3. *The maps $\pi_i \circ \tau : \mathscr{K}_2 \to \mathscr{K}_1$ are given by*

$$\begin{aligned}
\pi_1 \circ \tau\big((U_0 : U_1 : U_2 : U_3), (Z_0 : Z_1)\big) &= (U_0Z_0 - dU_3Z_1 : -U_0Z_1 + U_3Z_0), \\
\pi_2 \circ \tau\big((U_0 : U_1 : U_2 : U_3), (Z_0 : Z_1)\big) &= (U_0 : U_2) = (U_1 : U_3).
\end{aligned}$$

and $\pi_3 \circ \tau\big((U_0 : U_1 : U_2 : U_3), (Z_0 : Z_1)\big)$ is given by the equivalent expressions

$$\begin{aligned}
&\big((U_0^2 - dU_3^2)Z_0 \ : \ (U_0U_1 - U_2U_3)Z_0 + (U_0U_2 - dU_1U_3)Z_1\big) \\
&\big(-(U_0U_2 - U_1U_3)Z_0 + (U_0U_1 - dU_2U_3)Z_1 \ : \ (U_1^2 - U_2^2)Z_1\big)
\end{aligned}$$

7 Conclusion

The above polynomial maps for Montgomery endomorphism φ of $\mathscr{K}_2$ allows one to carry out a simultaneous symmetric addition and doubling on the Kummer surface. Besides the potential efficiency of this computation, this provides a simple geometric description of the basic ingredient for the Montgomery ladder on an Edwards model of an elliptic curve. The symmetry of the derived model for the split Kummer surface, and the endomorphisms ι, σ, and ρ provide the tools necessary for scalar multiplication on Edwards curves in cryptographic applications requiring protection from side channel attacks.

References

1. Bernstein, D.J., Lange, T.: Faster addition and doubling on elliptic curves. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 29–50. Springer, Heidelberg (2007)
2. Bernstein, D.J., Lange, T.: A complete set of addition laws for incomplete Edwards curves (2009), http://eprint.iacr.org/2009/580
3. Brier, E., Joye, M.: Weierstraß elliptic curves and side-channel attacks. In: Naccache, D., Paillier, P. (eds.) PKC 2002. LNCS, vol. 2274, pp. 335–345. Springer, Heidelberg (2002)
4. Edwards, H.: A normal form for elliptic curves. Bulletin of the American Mathematical Society 44, 393–422 (2007)
5. Hisil, H., Wong, K.K.-H., Carter, G., Dawson, E.: Twisted Edwards Curves Revisited. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 326–343. Springer, Heidelberg (2008)
6. Kohel, D.: Addition law structure of elliptic curves. Journal of Number Theory 131, 894–919 (2011), http://arxiv.org/abs/1005.3623
7. Izu, T., Takagi, T.: A Fast Parallel Elliptic Curve Multiplication Resistant against Side Channel Attacks. In: Naccache, D., Paillier, P. (eds.) PKC 2002. LNCS, vol. 2274, pp. 280–296. Springer, Heidelberg (2002)
8. Joye, M., Yen, S.-M.: The Montgomery Powering Ladder. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 291–302. Springer, Heidelberg (2003)
9. Lange, H., Ruppert, W.: Complete systems of addition laws on abelian varieties. Invent. Math. 79(3), 603–610 (1985)
10. Montgomery, P.: Speeding the Pollard and elliptic curve methods of factorization. Math. Comp. 48(177), 243–264 (1987)
11. Mumford, D.: On the equations defining abelian varieties I. Invent. Math. 1, 287–354 (1966)

A New Family of Quadriphase Sequences with Low Correlation

Jie Li[1], Xiangyong Zeng[1], and Lei Hu[2]

[1] The Faculty of Mathematics and Computer Science
Hubei University, Wuhan 430062, Hubei, China
xzeng@hubu.edu.cn
[2] The State Key Laboratory of Information Security
Graduate School of Chinese Academy of Sciences
Beijing 100049, China
hu@is.ac.cn

Abstract. For a positive integer n, a family of quadriphase sequences with period $4(2^n-1)$ is proposed. The correlation values of the family and their distribution are completely determined. The maximum non-trivial correlation magnitude is $4+2^{\frac{n+3}{2}}$ for odd n.

Keywords: Sequence, quadratic form, low correlation, linear span.

1 Introduction

Pseudorandom sequences have important applications in cryptography and communication systems. Low correlation is a favorable property of pseudorandom sequences. There are strong ties between sequences with low correlation and the theory of error-correcting codes, and by choosing cyclically inequivalent codewords from cyclic codes, families of sequences with low correlation can be efficiently constructed [3,6,12,16]. With the advantage of easy implementation, binary and quadriphase sequences are two classes of sequences most often used in practice. Further, from the Sidelnikov and Welch bounds [11,15], sometimes quadriphase sequences can have better correlation properties than binary sequences. However, compared with the binary case, the constructions of quadriphase sequences with desired properties are still not sufficient.

In 1992, two optimal quadriphase sequence families $\mathcal{A}$ and $\mathcal{B}$ were investigated in [1]. The family $\mathcal{A}$ has period 2^n-1 and family size 2^n+1, and the family $\mathcal{B}$ has period $2(2^n-1)$ and family size 2^{n-1}. Another optimal family $\mathcal{C}$ introduced in [14] has the same correlation properties as the family $\mathcal{B}$. Based on the families $\mathcal{B}$ and $\mathcal{C}$, Tang and Udaya obtained the family $\mathcal{D}$, which has period $2(2^n-1)$ and a double family size 2^n [13]. Recently, applying a generalized quadratic form, two optimal quadriphase sequence families $\mathcal{S}$ and $\mathcal{U}$ were proposed in [5], which have the same correlation properties as those of families $\mathcal{A}$ and $\mathcal{D}$ but larger linear spans, respectively. The exact correlation distribution of the family $\mathcal{U}$ was further investigated in [8].

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 246–262, 2011.
© Springer-Verlag Berlin Heidelberg 2011

In this paper, motivated by a construction of binary sequences with period $4(2^n-1)$ in [4], we propose a new family of quadriphase sequences with the same period. Based on the theory of $\mathbf{Z}_4$-valued quadratic forms [10], we calculate the correlation values of the proposed sequences and their distribution.

The remainder of this paper is organized as follows. Section 2 introduces some preliminary knowledge. In Section 3, we give some basic lemmas needed for our main results. The construction and correlation property of the new family are presented in Section 4. Section 5 concludes the study.

2 Preliminaries

2.1 Correlation Function

Let $u=\{u(t)\}_{t=0}^{L-1}$ and $u'=\{u'(t)\}_{t=0}^{L-1}$ be two quadriphase sequences of period L, and the *correlation function* $R_{u,u'}(\tau)$ between them at a shift $0\le\tau\le L-1$ is defined as

$$R_{u,u'}(\tau)=\sum_{t=0}^{L-1}\omega^{u(t)-u'(t+\tau)}$$

where $\omega=\sqrt{-1}$ is a primitive fourth complex root of unity. When $u=u'$, $R_{u,u'}(\tau)$ can be simplified as $R_u(\tau)$.

Let $\mathcal{S}$ be a family of M cyclically inequivalent quadriphase sequences of period L,

$$\mathcal{S}=\left\{u_i=\{u_i(t)\}_{t=0}^{L-1}:0\le i\le M-1\right\}\ .$$

The *maximum nontrivial correlation magnitude* $R_{\max}$ of $\mathcal{S}$ is

$$R_{\max}=\max\left\{|R_{u_i,u_j}(\tau)|:0\le i,j\le M-1 \text{ and } (i\ne j \text{ or } \tau\ne 0)\right\}\ .$$

2.2 Galois Ring

The Galois ring $\mathbf{R}=\mathrm{GR}(4,n)$ with 4^n elements and characteristic 4 denotes a Galois extension over $\mathbf{Z}_4$ of dimension n. As in the case of finite fields, the Galois ring $\mathbf{R}$ may be constructed as a quotient of the associated polynomial ring $\mathbf{Z}_4[x]$ [9].

Let $\bar{\cdot}$ denote the modulo 2 projection map from $\mathbf{Z}_4$ to $\mathbf{Z}_2$. Naturally, the projection map $\bar{\cdot}$ induces a homomorphism from $\mathbf{R}$ to the finite field $\mathbf{F}_{2^n}$ with 2^n elements. For convenience, the induced homomorphism is still denoted by $\bar{\cdot}$ in the sequel. A polynomial in $\mathbf{Z}_4[x]$ is called a *primitive basic irreducible polynomial* if its modulo 2 projection is a primitive irreducible polynomial in $\mathbf{Z}_2[x]$.

The multiplicative group in $\mathbf{R}$ contains an element β of order 2^n-1, then $\overline{\beta}$ is a primitive element of $\mathbf{F}_{2^n}$. In addition, the set $\mathcal{T}=\left\{0,1,\beta,\beta^2,\ldots,\beta^{2^n-2}\right\}$ is referred to as the *Teichmuller set*. Then, each $x\in\mathbf{R}$ can be uniquely written as

$$x=x_0+2x_1,\quad x_0,\,x_1\in\mathcal{T}\ . \tag{1}$$

The element $x \in \mathbf{R}$ is a *unit* if there is an element $y \in \mathbf{R}$ such that $xy = 1$. By (1), $x^2 = x_0^2$ and then x is a unit if and only if $x_0 \neq 0$. Since the addition operation in the Teichmuller set $\mathcal{T}$ is not closed, for x, $y \in \mathcal{T}$, we can define

$$x \oplus y = x + y + 2\sqrt{xy}$$

with $\sqrt{x} = x^{2^{n-1}}$. It is known that $(\mathcal{T}, \oplus, \cdot)$ is a finite field of size 2^n.

For any $y, z \in \mathcal{T}$, let $x = y + z + 2\sqrt{yz}$, then $x \in \mathcal{T}$. As (y, z) varies over $\mathcal{T} \times \mathcal{T}$, the pair (y, x) takes on every value in $\mathcal{T} \times \mathcal{T}$ precisely once. This fact was used in [7], and it will be applied in the proofs of the results in this paper.

The *trace function* $\mathrm{Tr}_1^n(\cdot)$ maps $\mathbf{R}$ to $\mathbf{Z}_4$. It is defined as

$$\mathrm{Tr}_1^n(x) = \sum_{i=0}^{n-1} \sigma^i(x),$$

where σ^i for $0 \leq i \leq n-1$ are the *Frobenius automorphisms* of $\mathbf{R}$ given by

$$\sigma^i(x_0 + 2x_1) = x_0^{2^i} + 2x_1^{2^i}, \quad \text{for any } x_0, x_1 \in \mathcal{T}\ .$$

Let $\mathrm{tr}_1^n(\cdot)$ denote the *trace function* from $\mathbf{F}_{2^n}$ to $\mathbf{F}_2$, i.e.,

$$\mathrm{tr}_1^n(x) = \sum_{i=0}^{n-1} x^{2^i}\ .$$

The following properties hold for any elements x, $y \in \mathbf{R}$:

P1) $2(x + y) = 2(\overline{x} + \overline{y})$;
P2) $\overline{2(xy)} = 2(\overline{x}\,\overline{y})$;
P3) $\overline{\mathrm{Tr}_1^n(x)} = \mathrm{tr}_1^n(\overline{x})$;
P4) $\mathrm{Tr}_1^n(x + y) = \mathrm{Tr}_1^n(x) + \mathrm{Tr}_1^n(y)$;
P5) $2\mathrm{Tr}_1^n(x) = 2\mathrm{Tr}_1^n\left(x^2\right)$.

2.3 $\mathbf{Z}_4$-Valued Quadratic Form

A symmetric bilinear form on $\mathcal{T}$ is a mapping $B : \mathcal{T} \times \mathcal{T} \longrightarrow \mathbf{Z}_2$ satisfying symmetry

$$B(x, y) = B(y, x)$$

and the bilinearity

$$B(ax \oplus by, z) = aB(x, z) + bB(y, z), \quad \text{for } a, b \in \mathbf{Z}_2\ .$$

In addition, B is called alternating if $B(x, x) = 0$ for all $x \in \mathcal{T}$. Otherwise, it is called nonalternating. The rank of B is defined as

$$\mathrm{rank}(B) = n - \dim_{\mathbf{Z}_2}(\mathrm{rad}(B))\ , \tag{2}$$

where $\mathrm{rad}(B) = \{x \in \mathcal{T} : B(x, y) = 0,\ \forall\ y \in \mathcal{T}\}$.

Definition 1 ([2]). *A $\mathbf{Z}_4$-valued quadratic form is a mapping $Q : \mathcal{T} \longrightarrow \mathbf{Z}_4$ that satisfies*

1) $Q(0) = 0$, and

2) $Q(x \oplus y) = Q(x) + Q(y) + 2B(x, y)$,

where $B : \mathcal{T} \times \mathcal{T} \longrightarrow \mathbf{Z}_2$ is the associated symmetric bilinear form of Q.

A $\mathbf{Z}_4$-valued quadratic form Q is similarly called alternating if its associated bilinear form is alternating. Otherwise, Q is called nonalternating. The rank of the quadratic form Q mentioned in Definition 1 is defined as $\mathrm{rank}(Q) = \mathrm{rank}(B)$.

For a nonalternating $\mathbf{Z}_4$-valued quadratic form Q: $\mathcal{T} \longrightarrow \mathbf{Z}_4$, the distribution of the values of the exponential sum

$$\chi_Q(b) = \sum_{x \in \mathcal{T}} \omega^{Q(x)+2\mathrm{Tr}_1^n(bx)}, \text{ where } b \text{ ranges over } \mathcal{T},$$

depends only on the rank of Q.

Lemma 1 ([10]). *Let Q be a nonalternating $\mathbf{Z}_4$-valued quadratic form of rank r, then the value distribution of the multiset $\{\chi_Q(b) : b \in \mathcal{T}\}$ is given by*

$$\chi_Q(b) = \begin{cases} 0, & 2^n - 2^r \text{ times}, \\ \pm(1+\omega)2^{n-\frac{r+1}{2}}, & 2^{r-2} \pm 2^{\frac{r-3}{2}} \text{ times}, \\ \pm(1-\omega)2^{n-\frac{r+1}{2}}, & 2^{r-2} \pm 2^{\frac{r-3}{2}} \text{ times} \end{cases}$$

for odd r, and

$$\chi_Q(b) = \begin{cases} 0, & 2^n - 2^r \text{ times}, \\ \pm 2^{n-\frac{r}{2}}, & 2^{r-2} \pm 2^{\frac{r}{2}-1} \text{ times}, \\ \pm\omega 2^{n-\frac{r}{2}}, & 2^{r-2} \text{ times (each)} \end{cases}$$

for even r.

Notations. The following notations are used throughout this paper:

- n is an integer, $n \geq 3$, and $s = \lfloor \frac{n}{2} \rfloor$;
- $\{\eta_0, \eta_1, \ldots, \eta_{2^n-1}\}$ is an enumeration of the elements in $\mathcal{T}$, and β is a generator of $\mathcal{T}$;
- $\alpha = \overline{\beta}$ is a primitive element of $\mathbf{F}_{2^n}$;
- For a subset $\mathcal{T}'$ of $\mathcal{T}$, set $\overline{\mathcal{T}'} = \{\overline{\eta} : \eta \in \mathcal{T}'\}$;
- $\mathrm{Re}(z)$ and $\mathrm{Im}(z)$ denote the real and imaginary parts of a complex number z, respectively;
- The symbol "+" in $x + y$ denotes the addition operator in $\mathbf{R}$ if $x, y \in \mathbf{R}$, and in $\mathbf{F}_{2^n}$ if $x, y \in \mathbf{F}_{2^n}$.

3 Basic Lemmas

This section gives several lemmas, which will be used to determine the correlation distribution of the quadriphase sequence family $\mathcal{S}$ proposed in Section 4.

Each unit $\gamma \in \mathbf{R}$ can be written as $\gamma = (1+2b)\beta^i$ for some i with $0 \leq i \leq 2^n - 2$ and for some $b \in \mathcal{T}$. Define an exponential sum

$$\xi(\gamma) = \sum_{x \in \mathcal{T}} \omega^{\mathrm{Tr}_1^n(\gamma x)} \ . \tag{3}$$

Then $\xi(\gamma) = \xi(1+2b)$ since $\mathcal{T} = \left\{\beta^i \zeta : \zeta \in \mathcal{T}\right\}$ for any fixed i. Without loss of generality, we can assume $\gamma = 1 + 2b$, and then $\xi(\gamma)$ can be rewritten as

$$\xi(\gamma) = \sum_{x \in \mathcal{T}} \omega^{\mathrm{Tr}_1^n((1+2b)x)} = \sum_{x \in \mathcal{T}} \omega^{Q(x)+2\mathrm{Tr}_1^n(bx)} \ ,$$

where $Q(x) = \mathrm{Tr}_1^n(x)$, $x \in \mathcal{T}$.

By Definition 1 and properties P4 and P5 of Subsection 2.2, one can verify that $Q(x) = \mathrm{Tr}_1^n(x)$ is a $\mathbf{Z}_4$-valued quadratic form and the associated bilinear form is $B(x,y) = \mathrm{tr}_1^n(\overline{xy})$. Then $Q(x) = \mathrm{Tr}_1^n(x)$ is nonalternating. By (2), the rank of $Q(x) = \mathrm{Tr}_1^n(x)$ is always n. Then from Lemma 1 we have the following result.

Lemma 2. *Let n be a positive integer, for any $0 \le i \le 2^n - 2$, the value distribution of sums $\xi\left((1+2b)\beta^i\right)$ is given as follows:*

1) if n is odd,

$$\xi\left((1+2b)\beta^i\right) = \begin{cases} 2^s \pm \omega 2^s, & 2^{2s-1} + 2^{s-1} \text{ times (each)} \ , \\ -2^s \pm \omega 2^s, & 2^{2s-1} - 2^{s-1} \text{ times (each)} \end{cases}$$

as b varies over $\mathcal{T}$.

2) if n is even,

$$\xi\left((1+2b)\beta^i\right) = \begin{cases} \pm 2^s, & 2^{n-2} \pm 2^{s-1} \text{ times} \ , \\ \pm\omega 2^s, & 2^{n-2} \text{ times (each)} \end{cases}$$

as b varies over $\mathcal{T}$.

The exponential sum $\xi(\gamma)$ has the following property, which will be used to study the correlation property of the family $\mathcal{S}$.

Lemma 3. *For $\delta \in \mathcal{T} \setminus \{0,1\}$ and $c \in \mathcal{T}$, let $\gamma_1 = 1 - \delta + 2c$ and γ_2 be a unit in $\mathbf{R}$. If $\gamma_2 - \gamma_1 = 2\delta$, then*

1) if n is odd,

$$\xi^2(\gamma_1) + \xi^2(\gamma_2) = \begin{cases} 0, & \text{if } \mathrm{tr}_1^n\left[\left(1+\overline{\delta}\right)^{-1}\right] = 0 \ , \\ \pm 2^{n+1}\omega, & \text{if } \mathrm{tr}_1^n\left[\left(1+\overline{\delta}\right)^{-1}\right] = 1 \ , \end{cases}$$

2) if n is even,

$$\xi^2(\gamma_1) + \xi^2(\gamma_2) = \begin{cases} \pm 2^{n+1}, & \text{if } \mathrm{tr}_1^n\left[\left(1+\overline{\delta}\right)^{-1}\right] = 0 \ , \\ 0, & \text{if } \mathrm{tr}_1^n\left[\left(1+\overline{\delta}\right)^{-1}\right] = 1 \ . \end{cases}$$

Proof. Notice that $1 - \overline{\delta} \neq 0$, consequently, $1 - \delta$ is a unit, and then it can be written as

$$1-\delta = a+2b' \tag{4}$$

for some $a \in \mathcal{T} \setminus \{0\}$ and $b' \in \mathcal{T}$. Thus $\overline{a} \neq 0$ and by (4), we have $\overline{a} = 1-\overline{\delta}$. Denote $b = c+b'$, then $\gamma_1 = a+2b$ and $\gamma_2 = a+2b+2\delta$. Thus we have

$$\begin{aligned}
&\xi^2(\gamma_1)+\xi^2(\gamma_2)\\
&= \sum_{x,y\in\mathcal{T}} \omega^{\mathrm{Tr}_1^n[(a+2b)(x+y)]} + \sum_{x,y\in\mathcal{T}} \omega^{\mathrm{Tr}_1^n[(a+2b+2\delta)(x+y)]}\\
&= \sum_{y,z\in\mathcal{T}} \omega^{\mathrm{Tr}_1^n\left[(a+2b)\left(2y+2\sqrt{yz}+z\right)\right]} + \sum_{y,z\in\mathcal{T}} \omega^{\mathrm{Tr}_1^n\left[(a+2b+2\delta)\left(2y+2\sqrt{yz}+z\right)\right]}\\
&= \sum_{y,z\in\mathcal{T}} \omega^{\mathrm{Tr}_1^n\left(2ay+2a\sqrt{yz}\right)+\mathrm{Tr}_1^n[(a+2b)z]} + \sum_{y,z\in\mathcal{T}} \omega^{\mathrm{Tr}_1^n\left(2ay+2a\sqrt{yz}\right)+\mathrm{Tr}_1^n[(a+2b+2\delta)z]}
\end{aligned}$$

where the substitution $x = y+z+2\sqrt{yz} \in \mathcal{T}$ is used in the second equal sign. By property P5 of Subsection 2.2, we have $\mathrm{Tr}_1^n\left(2a\sqrt{yz}\right) = \mathrm{Tr}_1^n\left(2a^2yz\right)$ and then

$$\begin{aligned}
&\xi^2(\gamma_1)+\xi^2(\gamma_2)\\
&= \sum_{y,z\in\mathcal{T}} \omega^{\mathrm{Tr}_1^n[2ay(1+az)]+\mathrm{Tr}_1^n[(a+2b)z]} + \sum_{y,z\in\mathcal{T}} \omega^{\mathrm{Tr}_1^n[2ay(1+az)]+\mathrm{Tr}_1^n[(a+2b+2\delta)z]}\\
&= \sum_{z\in\mathcal{T}} \omega^{\mathrm{Tr}_1^n[(a+2b)z]} \sum_{y\in\mathcal{T}} \omega^{2\mathrm{tr}_1^n[\overline{ay}(1+\overline{az})]} + \sum_{z\in\mathcal{T}} \omega^{\mathrm{Tr}_1^n[(a+2b+2\delta)z]} \sum_{y\in\mathcal{T}} \omega^{2\mathrm{tr}_1^n[\overline{ay}(1+\overline{az})]}
\end{aligned}$$

since $\mathrm{Tr}_1^n\left[2ay\left(1+az\right)\right] = 2\mathrm{tr}_1^n\left[\overline{ay}\left(1+\overline{az}\right)\right]$ by properties P1, P2, and P3 of Subsection 2.2.

Notice that

$$\sum_{y\in\mathcal{T}} \omega^{2\mathrm{tr}_1^n[\overline{ay}(1+\overline{az})]} = \sum_{\overline{y}\in\mathbf{F}_{2^n}} (-1)^{\mathrm{tr}_1^n[\overline{ay}(1+\overline{az})]} = \begin{cases} 0, & \text{if } 1+\overline{az} \neq 0\ , \\ 2^n, & \text{if } 1+\overline{az} = 0\ . \end{cases}$$

Since $\overline{a}, \overline{z} \in \mathbf{F}_{2^n}$, there is only one $\overline{z}$ satisfying $1+\overline{az} = 0$, for convenience, let $z' \in \mathcal{T}$ and $\overline{z'}$ be the solution of $1+\overline{az} = 0$. Thus $\overline{z'} = \overline{a}^{-1}$, and $\overline{\mathrm{Tr}_1^n\left[(a+2b)\,z'\right]} = \mathrm{tr}_1^n\left(\overline{a}\overline{z'}\right) = \mathrm{tr}_1^n(1)$ by property P3.

1) When n is odd, we have $\mathrm{tr}_1^n(1) = 1$, this shows $\mathrm{Tr}_1^n\left[(a+2b)\,z'\right] = 1$ or 3. The above analysis implies that

$$\begin{aligned}
\xi^2(\gamma_1)+\xi^2(\gamma_2) &= 2^n\omega^{\mathrm{Tr}_1^n\left[(a+2b)z'\right]}\left(1+\omega^{2\mathrm{Tr}_1^n\left(\overline{\delta z'}\right)}\right)\\
&= 2^n\omega^{\mathrm{Tr}_1^n\left[(a+2b)z'\right]}\left(1+(-1)^{\mathrm{tr}_1^n\left(\overline{\delta}\overline{a}^{-1}\right)}\right)\\
&= 2^n\omega^{\mathrm{Tr}_1^n\left[(a+2b)z'\right]}\left(1+(-1)^{1+\mathrm{tr}_1^n\left[\left(1+\overline{\delta}\right)^{-1}\right]}\right)\\
&= \begin{cases} 0, & \text{if } \mathrm{tr}_1^n\left[\left(1+\overline{\delta}\right)^{-1}\right] = 0\ , \\ \pm 2^{n+1}\omega, & \text{if } \mathrm{tr}_1^n\left[\left(1+\overline{\delta}\right)^{-1}\right] = 1 \end{cases}
\end{aligned}$$

since $\mathrm{tr}_1^n\left(\overline{\delta}\overline{a}^{-1}\right) = \mathrm{tr}_1^n\left[\left(1+\overline{a}\right)\overline{a}^{-1}\right] = \mathrm{tr}_1^n(1)+\mathrm{tr}_1^n\left(\overline{a}^{-1}\right) = 1+\mathrm{tr}_1^n\left[\left(1+\overline{\delta}\right)^{-1}\right]$.

2) When n is even, we have $\mathrm{tr}_1^n(1) = 0$, this shows $\mathrm{Tr}_1^n\left[(a+2b)\,z'\right] = 0$ or 2. We similarly have

$$\xi^2(\gamma_1) + \xi^2(\gamma_2) = \begin{cases} \pm 2^{n+1}, \text{ if } \mathrm{tr}_1^n\left[\left(1+\overline{\delta}\right)^{-1}\right] = 0 \ , \\ 0, \qquad \text{if } \mathrm{tr}_1^n\left[\left(1+\overline{\delta}\right)^{-1}\right] = 1 \ . \end{cases}$$

□

Lemma 4. *Let $\theta \in \mathbf{R}$ and $\eta \in \mathcal{T}$. When η runs through $\mathcal{T}$, $2(\eta+\theta)$ runs through $2\mathcal{T} = \{2\delta : \delta \in \mathcal{T}\}$.*

Proof. For two distinct elements η and η' of $\mathcal{T}$, let $\eta + \theta = a + 2b$ and $\eta' + \theta = a' + 2b'$ where a, b, a' and $b' \in \mathcal{T}$. Then $2(\eta+\theta) = 2a$ and $2(\eta'+\theta) = 2a'$. If $a = a'$, then $2\eta = 2\eta'$. Since each $x \in \mathbf{R}$ can be uniquely written as (1), this shows that $0 + 2\eta = 0 + 2\eta' \in \mathbf{R}$ implies $\eta = \eta'$, which contradicts with $\eta \neq \eta'$. Thus $a \neq a'$ and then $2(\eta+\theta) = 2a$ runs through $2\mathcal{T}$ when η runs through $\mathcal{T}$. □

Lemma 5. *There is a subset $\mathcal{T}_0$ with 2^{n-1} elements of $\mathcal{T}$ such that*

$$\mathcal{T} = \mathcal{T}_0 \bigcup \{1 + \eta + 2\sqrt{\eta} : \eta \in \mathcal{T}_0\} \ .$$

Proof. Since $\overline{\mathcal{T}} = \mathbf{F}_{2^n}$, we have $\overline{\eta_i} \neq 1 + \overline{\eta_i}$ for any $0 \leq i < 2^n$. Thus $\mathbf{F}_{2^n}$ can be written as the union of two disjoint subsets A_0 and A_1, i.e.,

$$A_0 = \{\overline{\eta_i} : i \in I\}, \text{ and } A_1 = \{1 + \overline{\eta_i} : i \in I\}$$

where I is a subset of $\{0, 1, \cdots, 2^n - 1\}$ and contains 2^{n-1} integers. Let $\mathcal{T}_0$ be a subset of $\mathcal{T}$ such that $\overline{\mathcal{T}_0} = A_0$. Then $\overline{\mathcal{T} \backslash \mathcal{T}_0} = A_1$.

For any $\eta \in \mathcal{T}_0$, we have $\overline{\eta} \in A_0$ and then $1 + \overline{\eta} \in A_1$. Let $1 + \sqrt{\eta} = a + 2b$ for some $a, b \in \mathcal{T}$, then $1 + \eta + 2\sqrt{\eta} = (1+\sqrt{\eta})^2 = a^2 \in \mathcal{T}$. Since $1 + \eta + 2\sqrt{\eta} \in \mathcal{T}$ and $\overline{1 + \eta + 2\sqrt{\eta}} = 1 + \overline{\eta} \in A_1$, we have $1 + \eta + 2\sqrt{\eta} \in \overline{\mathcal{T} \backslash \mathcal{T}_0}$. □

4 A New Family of Quadriphase Sequences and Their Correlation Distribution

In this section, we define a new family of quadriphase sequences with period $4(2^n - 1)$ and determine its correlation distribution.

Let the notation $l/4$ represent $2^{n-2} \cdot l$ in the index positions of the exponentiations.

Definition 2. *Define a quadriphase sequence family $\mathcal{S} = \{u_0, u_1, \ldots, u_{2^n-1}\}$, where $u_i = \{u_i(t)\}_{t=0}^{4(2^n-1)-1}$ is given by*

$$u_i(t) = \begin{cases} \mathrm{Tr}_1^n\left(\left(1+2\eta_i\right)\left(1+2v_0\right)\beta^{t_0}\right) + 2, & t = 4t_0 \ , \\ \mathrm{Tr}_1^n\left(\left(1+2\eta_i\right)\left(1+2v_1\right)\beta^{t_0+1/4}\right), & t = 4t_0+1 \ , \\ \mathrm{Tr}_1^n\left(\left(1+2\eta_i\right)\left(1+2v_2\right)\beta^{t_0+2/4}\right), & t = 4t_0+2 \ , \\ \mathrm{Tr}_1^n\left(\left(1+2\eta_i\right)\left(1+2v_3\right)\beta^{t_0+3/4}\right), & t = 4t_0+3 \end{cases}$$

with $v_0 = v_3 = 0$ and $v_1 = v_2 = 1$.

In the sequel, we will study the correlation distribution of the sequence family $\mathcal{S}$.

For fixed i and j and two changeable indexes l and k with $0 \le l, k < 4$, define

$$\delta_l = \beta^{\tau_0 + l/4}, \quad \text{and} \quad \gamma_{l,k} = 1 + 2(\eta_i + v_k) - (1 + 2(\eta_j + v_{k+l}))\,\delta_l \tag{5}$$

where $0 \le \tau_0 < 2^n - 1$ and the subscript $k + l$ is taken modulo 4.

Notice that in $\mathbf{R}$

$$\begin{aligned} & 1 + 2(\eta_i + v_k + 1) - (1 + 2(\eta_j + v_{k+l} + 1))\,\delta_l \\ &= 3 + 2(\eta_i + v_k) - (3 + 2(\eta_j + v_{k+l}))\,\delta_l \\ &= -1 + 2(\eta_i + v_k) - (-1 + 2(\eta_j + v_{k+l}))\,\delta_l \\ &= -[1 - 2(\eta_i + v_k) - (1 - 2(\eta_j + v_{k+l}))\,\delta_l] \\ &= -[1 + 2(\eta_i + v_k) - (1 + 2(\eta_j + v_{k+l}))\,\delta_l] \ , \end{aligned} \tag{6}$$

and

$$\begin{aligned} & 1 + 2(\eta_i + v_k + 1) - (1 + 2(\eta_j + v_{k+l}))\,\delta_l \\ &= -[1 + 2(\eta_i + v_k) - (1 + 2(\eta_j + v_{k+l} + 1))\,\delta_l] \ . \end{aligned} \tag{7}$$

Consequently, by (5), (6) and (7) we have

$$\gamma_{l,k} = \begin{cases} -\gamma_{0,1}, \text{ if } l = k = 0 \ , \\ -\gamma_{l,k'}, \text{ if } l \text{ is odd and } k \ne k',\ k + k' \text{ is even} \ , \\ \gamma_{l,k'}, \quad \text{if } l \text{ is even and } k + k' = 3 \end{cases} \tag{8}$$

for any $0 \le l, k \ne k' < 4$.

By (8), we have

$$\xi(\gamma_{l,k}) = \begin{cases} \xi^*(\gamma_{0,1}), \text{ if } l = k = 0 \ , \\ \xi^*(\gamma_{l,k'}), \text{ if } l \text{ is odd and } k \ne k',\ k + k' \text{ is even} \ , \\ \xi(\gamma_{l,k'}), \quad \text{if } l \text{ is even and } k + k' = 3 \end{cases} \tag{9}$$

for any $0 \le l, k \ne k' < 4$, where $\xi^*(\gamma_{l,k'})$ denotes the complex conjugate of $\xi(\gamma_{l,k'})$.

By (5), we have $\overline{\gamma_{l,k}} = 1 - \overline{\delta_l}$ and then $\gamma_{l,k}$ is a unit for any $0 \le l, k \le 3$ if $\delta_l \ne 1$. In this case, by Lemma 2 we have $\xi(\gamma_{l,k}) = \pm 2^s(1 \pm \omega)$ when n is odd, while $\xi(\gamma_{l,k}) = \pm 2^s\omega$ or $\pm 2^s$ when n is even.

Lemma 6. *For $l \in \{1, 3\}$ and $\delta_l \ne 1$, for any fixed j and δ_l, let $K_{1,i} = -\xi(\gamma_{1,0}) + \xi(\gamma_{1,1}) + \xi(\gamma_{1,2}) - \xi(\gamma_{1,3})$ and $K_{3,i} = -\xi(\gamma_{3,0}) - \xi(\gamma_{3,1}) + \xi(\gamma_{3,2}) + \xi(\gamma_{3,3})$. Then for any complex number ε, we have*

$$|\{i : K_{l,i} = \varepsilon, 0 \le i < 2^n\}| = |\{i : K_{l,i} = -\varepsilon, 0 \le i < 2^n\}| \ .$$

Proof. For η_i and $\eta_r \in \mathcal{T}$ satisfying $\eta_r = 1 + \eta_i + 2\sqrt{\eta_i}$, a direct verification shows $K_{l,i} = -K_{l,r}$ for any $l \in \{1, 3\}$. By Lemma 5, we have

$$|\{i : K_{l,i} = \varepsilon, 0 \le i < 2^n\}| = |\{i : K_{l,i} = -\varepsilon, 0 \le i < 2^n\}|$$

for any complex number ε. □

Lemma 7. *For $l \in \{1,3\}$ and $\delta_l \neq 1$, we have* $\sum_{i=0}^{2^n-1} \xi(\gamma_{l,1})\xi(\gamma_{l,2}) = 0$.

Proof. Notice that $1-\delta_l$ is a unit, then it can be represented as

$$1-\delta_l = a + 2b'$$

for some $a \in \mathcal{T}\setminus\{0\}$ and $b' \in \mathcal{T}$. Thus $1-\overline{\delta_l} = \overline{a}$. Let $b = 1+\eta_i-\eta_j\delta_l+b'-v_{l+1}\delta_l$, then

$$\gamma_{l,1} = 1-\delta_l+2\left(1+\eta_i-\eta_j\delta_l-v_{l+1}\delta_l\right) = a+2b$$

and

$$\gamma_{l,2} = \gamma_{l,1}+2\delta_l = a+2b+2\delta_l \ .$$

By Lemma 4, when i ranges from 0 to 2^n-1, $2b$ runs through $2\mathcal{T}$. Thus, we have

$$\begin{aligned}
&\sum_{i=0}^{2^n-1} \xi(\gamma_{l,1})\xi(\gamma_{l,2}) \\
&= \sum_{i=0}^{2^n-1} \sum_{x\in\mathcal{T}} \omega^{\mathrm{Tr}_1^n(\gamma_{l,1}x)} \sum_{y\in\mathcal{T}} \omega^{\mathrm{Tr}_1^n(\gamma_{l,2}y)} \\
&= \sum_{2b\in 2\mathcal{T}} \sum_{x,y\in\mathcal{T}} \omega^{\mathrm{Tr}_1^n[(a+2b)(x+y)+2\delta_l y]} \\
&= \sum_{x,y\in\mathcal{T}} \omega^{\mathrm{Tr}_1^n(a(x+y)+2\delta_l y)} \sum_{b\in\mathcal{T}} \omega^{\mathrm{Tr}_1^n[2b(x+y)]} \\
&= 2^n \sum_{x\in\mathcal{T}} \omega^{\mathrm{Tr}_1^n[2(a+\delta_l)x]} \\
&= 2^n \sum_{\overline{x}\in\mathbf{F}_{2^n}} \omega^{2\mathrm{tr}_1^n\left[\left(\overline{a}+\overline{\delta_l}\right)\overline{x}\right]} \\
&= 2^n \sum_{\overline{x}\in\mathbf{F}_{2^n}} (-1)^{\mathrm{tr}_1^n(\overline{x})} \\
&= 0 \ .
\end{aligned} \tag{10}$$

□

With the above preparations, we will determine the correlation distribution of the family $\mathcal{S}$ in Subsections 4.1 and 4.2 for odd and even n, respectively.

4.1 Correlation Distribution of the Family $\mathcal{S}$ for Odd n

In this subsection, we always assume that n is odd.

Theorem 1. *The family $\mathcal{S}$ has correlation distribution as*

$$R_{u_i,u_j}(\tau) = \begin{cases} 4(2^n-1), & 2^n \textit{ times} \ , \\ -4, & 2^n(2^n-1) \textit{ times} \ , \\ 0, & 2^{2n}(2^{n+1}-1) \textit{ times} \ , \\ -4\pm 2^{\frac{n+3}{2}}, & 2^n(2^n-2)(2^{n-1}\pm 2^{\frac{n-1}{2}}) \textit{ times} \ , \\ \pm\omega 2^{\frac{n+3}{2}}, & 2^{2n}(2^{n-1}-1) \textit{ times (each)} \ . \end{cases} \tag{11}$$

Proof. For any $0 \le \tau < 4(2^n-1)$, it can be written as $\tau = 4\tau_0 + l$ for $0 \le \tau_0 < 2^n - 1$ and $0 \le l < 4$. From the definition of δ_l and $\gamma_{l,k}$ in (5), the correlation function between the sequences u_i and u_j in $\mathcal{S}$ is

$$R_{u_i,u_j}(\tau) = \begin{cases} \xi(\gamma_{0,0}) + \xi(\gamma_{0,1}) + \xi(\gamma_{0,2}) + \xi(\gamma_{0,3}) - 4, & l = 0 \ , \\ -\xi(\gamma_{1,0}) + \xi(\gamma_{1,1}) + \xi(\gamma_{1,2}) - \xi(\gamma_{1,3}), & l = 1 \ , \\ -\xi(\gamma_{2,0}) + \xi(\gamma_{2,1}) - \xi(\gamma_{2,2}) + \xi(\gamma_{2,3}), & l = 2 \ , \\ -\xi(\gamma_{3,0}) - \xi(\gamma_{3,1}) + \xi(\gamma_{3,2}) + \xi(\gamma_{3,3}), & l = 3 \ . \end{cases} \tag{12}$$

The analysis of the correlation distribution can proceed according to five cases as below.

i) When $l = 0$ and $\delta_l = 1$, we have $\tau_0 = 0$. Notice that for any $0 \le k \le 3$, we have

$$\xi(\gamma_{0,k}) = \sum_{x \in \mathcal{T}} \omega^{\mathrm{Tr}_1^n[2(\eta_i - \eta_j)x]} = \begin{cases} 2^n, \ \eta_i = \eta_j \ , \\ 0, \ \eta_i \ne \eta_j \ . \end{cases}$$

As a consequence,

$$R_{u_i,u_j}(\tau) = \begin{cases} 4(2^n-1), & 2^n \text{ times} \ , \\ -4, & 2^n(2^n-1) \text{ times} \end{cases}$$

as i and j vary from 0 to $2^n - 1$.

ii) When $l \in \{1, 3\}$ and $\delta_l = 1$, by (5) we have

$$\gamma_{1,0} = 1 + 2\eta_i - (1 + 2(\eta_j + 1)) = 1 + 2(\eta_i + 1) - (1 + 2\eta_j) = \gamma_{1,2} \ .$$

Similarly, we have $\gamma_{3,0} = \gamma_{3,2}$ and $\gamma_{l,1} = \gamma_{l,3}$ for $l \in \{1, 3\}$. By (12), $R_{u_i,u_j}(\tau) = 0$ occurs 2^{2n} times as i and j vary from 0 to $2^n - 1$.

iii) When $l = 2$, by (9) we have $R_{u_i,u_j}(\tau) = 0$ occurs $2^{2n}(2^n - 1)$ times as i, j vary from 0 to $2^n - 1$, and τ_0 varies from 0 to $2^n - 2$.

iv) When $l = 0$ and $\delta_l \ne 1$, by (9) and (12) we have $R_{u_i,u_j}(\tau) = 4\mathrm{Re}\,[\xi(\gamma_{0,0})] - 4$. Let $1 - \delta_0 = \beta^t(1 + 2a)$ for some integer $1 \le t < 2^n - 1$ and some $a \in \mathcal{T}$, then

$$\gamma_{0,0} = 1 + 2\eta_i - (1 + 2\eta_j)\delta_0 = \beta^t(1 + 2b)$$

where $b = a + \eta_i\beta^{-t} + \eta_j\delta_0\beta^{-t} \in \mathbf{R}$. For any fixed $\delta_0 \ne 1$ and any fixed j, when i ranges from 0 to $2^n - 1$, $2b$ runs through $2\mathcal{T}$ by Lemma 4.

By Lemma 2, we have that $R_{u_i,u_j}(\tau) = -4 \pm 2^{s+2}$ occurs $2^n(2^n - 2)(2^{2s} \pm 2^s)$ times as i, j range from 0 to $2^n - 1$, and τ_0 varies from 1 to $2^n - 2$.

v) When $l \in \{1, 3\}$ and $\delta_l \ne 1$, by (9) and Lemma 2, we have $R_{u_i,u_j}(\tau) \in \{0, \pm 2^{s+2}\omega\}$.

By Lemma 3, we have

$$\xi^2(\gamma_{l,1}) + \xi^2(\gamma_{l,2}) = \begin{cases} 0, & \text{if } \ \mathrm{tr}_1^n\left[\left(1 + \overline{\delta_l}\right)^{-1}\right] = 0 \ , \\ \pm 2^{n+1}\omega, & \text{if } \ \mathrm{tr}_1^n\left[\left(1 + \overline{\delta_l}\right)^{-1}\right] = 1 \ . \end{cases} \tag{13}$$

By $\delta_l \in \mathcal{T}$ and $\delta_l \ne 0, 1$, we have $\overline{\delta_l} \in \mathbf{F}_{2^n} \setminus \{0, 1\}$. Since $\mathrm{tr}_1^n(0) = 0$ and $\mathrm{tr}_1^n(1) = 1$, by the balance property of trace function we have that both the equations $\mathrm{tr}_1^n(x) = 0$ and $\mathrm{tr}_1^n(x) = 1$ have $2^{n-1} - 1$ solutions in $\mathbf{F}_{2^n} \setminus \{0, 1\}$.

(1) For any fixed $\delta_l \in \mathcal{T} \setminus \{0,1\}$ satisfying $\mathrm{tr}_1^n\left[\left(1+\overline{\delta_l}\right)^{-1}\right] = 0$ and for any fixed $0 \le j \le 2^n - 1$, we discuss the distribution of the pair $(\xi(\gamma_{l,1}), \xi(\gamma_{l,2}))$ when i varies from 0 to $2^n - 1$.

In the case of $l = 1$, by (13) the pair $(\xi(\gamma_{1,1}), \xi(\gamma_{1,2}))$ takes at most 8 values as in Table 1.

Table 1. The distribution of the pair $(\xi(\gamma_{1,1}), \xi(\gamma_{1,2}))$ when $\mathrm{tr}_1^n\left[\left(1+\overline{\delta_1}\right)^{-1}\right] = 0$

$(\xi(\gamma_{1,1}), \xi(\gamma_{1,2}))$	$\mathrm{Im}(\xi(\gamma_{1,1}) + \xi(\gamma_{1,2}))$	$\xi(\gamma_{1,1})\xi(\gamma_{1,2})$	Frequency
$(2^s + \omega 2^s, 2^s - \omega 2^s)$	0	2^n	x_1
$(2^s + \omega 2^s, -2^s + \omega 2^s)$	2^{s+1}	-2^n	x_2
$(2^s - \omega 2^s, 2^s + \omega 2^s)$	0	2^n	x_3
$(2^s - \omega 2^s, -2^s - \omega 2^s)$	-2^{s+1}	-2^n	x_4
$(-2^s + \omega 2^s, -2^s - \omega 2^s)$	0	2^n	x_5
$(-2^s + \omega 2^s, 2^s + \omega 2^s)$	2^{s+1}	-2^n	x_6
$(-2^s - \omega 2^s, -2^s + \omega 2^s)$	0	2^n	x_7
$(-2^s - \omega 2^s, 2^s - \omega 2^s)$	-2^{s+1}	-2^n	x_8

By Lemma 2, we have

$$\begin{cases} x_1 + x_2 = x_3 + x_4 = x_1 + x_8 = x_3 + x_6 = 2^{2s-1} + 2^{s-1} \\ x_5 + x_6 = x_7 + x_8 = x_2 + x_7 = x_4 + x_5 = 2^{2s-1} - 2^{s-1} \end{cases},$$

Thus $x_2 = x_8$ and $x_4 = x_6$. By Lemma 7, we also have

$$2^n(x_1 - x_2 + x_3 - x_4 + x_5 - x_6 + x_7 - x_8) = 0 \ .$$

The above analysis together with $x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 = 2^n$ gives

$$\begin{cases} x_1 + x_3 + x_5 + x_7 = 2^{2s} \\ x_2 + x_6 = x_4 + x_8 = 2^{2s-1} \end{cases}.$$

Therefore,

$$R_{u_i,u_j}(\tau) = 2\omega \mathrm{Im}(\xi(\gamma_{1,1}) + \xi(\gamma_{1,2})) = \begin{cases} 0, & 2^n(2^{n-1}-1)2^{2s} \text{ times} \\ \pm\omega 2^{s+2}, & 2^n(2^{n-1}-1)2^{2s-1} \text{times (each)} \end{cases},$$

as i and j vary from 0 to $2^n - 1$, δ_1 runs through $\mathcal{T} \setminus \{0,1\}$ and satisfies $\mathrm{tr}_1^n\left[\left(1+\overline{\delta_1}\right)^{-1}\right] = 0$.

In the case of $l = 3$, we similarly have that the pair $(\xi(\gamma_{3,1}), \xi(\gamma_{3,2}))$ takes at most 8 values as in Table 2.

By a similar analysis as in the case of $l = 1$, we have

$$\begin{cases} x_1 + x_3 + x_5 + x_7 = 2^{2s} \\ x_2 + x_4 + x_6 + x_8 = 2^{2s} \end{cases}.$$

By Lemma 6, we have $x_1 + x_5 = x_3 + x_7 = 2^{2s-1}$.

Table 2. The distribution of the pair $(\xi(\gamma_{3,1}), \xi(\gamma_{3,2}))$ when $\mathrm{tr}_1^n\left[\left(1+\overline{\delta_3}\right)^{-1}\right]=0$

$(\xi(\gamma_{3,1}), \xi(\gamma_{3,2}))$	$\mathrm{Im}(\xi(\gamma_{3,2})-\xi(\gamma_{3,1}))$	$\xi(\gamma_{3,1})\xi(\gamma_{3,2})$	Frequency
$(2^s+\omega 2^s, 2^s-\omega 2^s)$	-2^{s+1}	2^n	x_1
$(2^s+\omega 2^s, -2^s+\omega 2^s)$	0	-2^n	x_2
$(2^s-\omega 2^s, 2^s+\omega 2^s)$	2^{s+1}	2^n	x_3
$(2^s-\omega 2^s, -2^s-\omega 2^s)$	0	-2^n	x_4
$(-2^s+\omega 2^s, -2^s-\omega 2^s)$	-2^{s+1}	2^n	x_5
$(-2^s+\omega 2^s, 2^s+\omega 2^s)$	0	-2^n	x_6
$(-2^s-\omega 2^s, -2^s+\omega 2^s)$	2^{s+1}	2^n	x_7
$(-2^s-\omega 2^s, 2^s-\omega 2^s)$	0	-2^n	x_8

Therefore, we have

$$R_{u_i,u_j}(\tau)=2\omega\mathrm{Im}(\xi(\gamma_{3,2})-\xi(\gamma_{3,1}))=\begin{cases}0, & 2^{2n-1}(2^{n-1}-1) \text{ times }\\ \pm\omega 2^{s+2}, & 2^{2n-2}(2^{n-1}-1) \text{ times (each)}\end{cases},$$

as i, j vary from 0 to 2^n-1, and δ_3 runs through $\mathcal{T}\setminus\{0,1\}$ and satisfies $\mathrm{tr}_1^n\left[\left(1+\overline{\delta_3}\right)^{-1}\right]=0$.

(2) For any fixed $\delta_l\in\mathcal{T}\setminus\{0,1\}$ satisfying $\mathrm{tr}_1^n\left[\left(1+\overline{\delta_l}\right)^{-1}\right]=1$ and for any fixed $0\le j\le 2^n-1$, we discuss the distribution of the pair $(\xi(\gamma_{l,1}), \xi(\gamma_{l,2}))$ when i varies from 0 to 2^n-1.

In the case of $l=1$, by (13), the pair $(\xi(\gamma_{1,1}), \xi(\gamma_{1,2}))$ takes at most 8 values as in Table 3.

Table 3. The distribution of the pair $(\xi(\gamma_{1,1}), \xi(\gamma_{1,2}))$ when $\mathrm{tr}_1^n\left[\left(1+\overline{\delta_1}\right)^{-1}\right]=1$

$(\xi(\gamma_{1,1}), \xi(\gamma_{1,2}))$	$\mathrm{Im}(\xi(\gamma_{1,1})+\xi(\gamma_{1,2}))$	Frequency
$(2^s+\omega 2^s, 2^s+\omega 2^s)$	2^{s+1}	y_1
$(2^s+\omega 2^s, -2^s-\omega 2^s)$	0	y_2
$(2^s-\omega 2^s, 2^s-\omega 2^s)$	-2^{s+1}	y_3
$(2^s-\omega 2^s, -2^s+\omega 2^s)$	0	y_4
$(-2^s+\omega 2^s, -2^s+\omega 2^s)$	2^{s+1}	y_5
$(-2^s+\omega 2^s, 2^s-\omega 2^s)$	0	y_6
$(-2^s-\omega 2^s, -2^s-\omega 2^s)$	-2^{s+1}	y_7
$(-2^s-\omega 2^s, 2^s+\omega 2^s)$	0	y_8

By (9), we have $R_{u_i,u_j}(\tau)=2\omega\mathrm{Im}(\xi(\gamma_{1,1})+\xi(\gamma_{1,2}))$. Further, we have

$$\begin{cases}\gamma_{1,1}=1+2\eta_i+2-(1+2\eta_j+2)\delta_1 \ ,\\ \gamma_{1,2}=1+2\eta_i+2-(1+2\eta_j)\delta_1 \ .\end{cases} \tag{14}$$

In the case of $l = 3$, we similarly have $R_{u_i,u_j}(\tau) = 2\omega \mathrm{Im}(\xi(\gamma_{3,2}) - \xi(\gamma_{3,1}))$ and

$$\begin{cases} \gamma_{3,1} = 1 + 2\eta_i + 2 - (1 + 2\eta_j)\delta_3 \ , \\ \gamma_{3,2} = 1 + 2\eta_i + 2 - (1 + 2\eta_j + 2)\delta_3 \ . \end{cases} \tag{15}$$

By (14) and (15), we have that the pair $(\xi(\gamma_{1,1}), \xi(\gamma_{1,2}))$ has the same distribution with $(\xi(\gamma_{3,2}), \xi(\gamma_{3,1}))$, whose distribution is given by Table 4.

Table 4. The distribution of the pair $(\xi(\gamma_{3,2}), \xi(\gamma_{3,1}))$ when $\mathrm{tr}_1^n\left[\left(1 + \overline{\delta_3}\right)^{-1}\right] = 1$

$(\xi(\gamma_{3,2}), \xi(\gamma_{3,1}))$	$\mathrm{Im}(\xi(\gamma_{3,2}) - \xi(\gamma_{3,1}))$	Frequency
$(2^s + \omega 2^s, 2^s + \omega 2^s)$	0	y_1
$(2^s + \omega 2^s, -2^s - \omega 2^s)$	2^{s+1}	y_2
$(2^s - \omega 2^s, 2^s - \omega 2^s)$	0	y_3
$(2^s - \omega 2^s, -2^s + \omega 2^s)$	-2^{s+1}	y_4
$(-2^s + \omega 2^s, -2^s + \omega 2^s)$	0	y_5
$(-2^s + \omega 2^s, 2^s - \omega 2^s)$	2^{s+1}	y_6
$(-2^s - \omega 2^s, -2^s - \omega 2^s)$	0	y_7
$(-2^s - \omega 2^s, 2^s + \omega 2^s)$	-2^{s+1}	y_8

Notice that

$$y_1 + y_2 + y_3 + y_4 + y_5 + y_6 + y_7 + y_8 = 2^n \ .$$

By Lemma 6, and Tables 3, 4, we have the distribution of the multi-set

$$\left\{ R_{u_i,u_j}(\tau) : 0 \le i, j < 2^n, l \in \{1, 3\}, \mathrm{tr}_1^n\left[\left(1 + \overline{\delta_l}\right)^{-1}\right] = 1 \right\}$$

as

$$R_{u_i,u_j}(\tau) = \begin{cases} 0, & 2^{2n}(2^{n-1} - 1) \text{ times} \ , \\ \pm\omega 2^{s+2}, & 2^{2n-1}(2^{n-1} - 1) \text{ times (each)} \ . \end{cases}$$

Combining the above five cases, the correlation distribution of the family $\mathcal{S}$ is completely determined as (11). □

Example 1. For $n = 5$, we use the primitive basic irreducible polynomial $f(x) = x^5 + 2x^4 + x^3 + 3$ in $\mathbf{Z}_4[x]$ to generate the Galois ring $\mathbf{R} = \mathrm{GR}(4, 5)$. By computer computation, the family $\mathcal{S}$ defined in Definition 2 can be verified to have the correlation distribution as in Table 5, which is consistent with Theorem 1.

Table 5. The correlation distribution of the family $\mathcal{S}$ for $n = 5$

Value	124	-4	0	-20	12	-16ω	16ω
Frequency	32	992	64512	11520	19200	15360	15360

4.2 Correlation Distribution of the Family $\mathcal{S}$ for Even n

In this subsection, we always assume that n is even.

Theorem 2. *The family $\mathcal{S}$ has correlation distribution as*

$$R_{u_i,u_j}(\tau)=\begin{cases}4(2^n-1), & 2^n \text{ times },\\ -4, & 2^n(2^{2n-1}-1) \text{ times },\\ 0, & 2^{2n}(7\cdot 2^{n-2}-2) \text{ times },\\ -4\pm 2^{\frac{n}{2}+2}, & 2^n(2^n-2)(2^{n-2}\pm 2^{\frac{n}{2}-1}) \text{ times },\\ \pm\omega 2^{\frac{n}{2}+1}, & 2^{3n-1} \text{ times (each) },\\ \pm\omega 2^{\frac{n}{2}+2}, & 2^{2n-1}(2^{n-2}-1) \text{ times (each) }.\end{cases} \tag{16}$$

Proof. For any $0\le\tau<4(2^n-1)$, we write $\tau=4\tau_0+l$ where $0\le\tau_0<2^n-1$ and $0\le l<4$. Then the correlation function between the sequences u_i and u_j in $\mathcal{S}$ is

$$R_{u_i,u_j}(\tau)=\begin{cases}\xi(\gamma_{0,0})+\xi(\gamma_{0,1})+\xi(\gamma_{0,2})+\xi(\gamma_{0,3})-4, & l=0,\\ -\xi(\gamma_{1,0})+\xi(\gamma_{1,1})+\xi(\gamma_{1,2})-\xi(\gamma_{1,3}), & l=1,\\ -\xi(\gamma_{2,0})+\xi(\gamma_{2,1})-\xi(\gamma_{2,2})+\xi(\gamma_{2,3}), & l=2,\\ -\xi(\gamma_{3,0})-\xi(\gamma_{3,1})+\xi(\gamma_{3,2})+\xi(\gamma_{3,3}), & l=3.\end{cases} \tag{17}$$

The analysis of the correlation distribution can proceed according to five cases in a similar way as in Theorem 1. Here we only analyze the case when $l\in\{1,3\}$ and $\delta_l\ne 1$, since the other cases can be similarly analyzed as Theorem 1.

When $l\in\{1,3\}$ and $\delta_l\ne 1$, By (9) and Lemma 2, we have $R_{u_i,u_j}(\tau)\in\{0,\pm 2^{s+1}\omega,\pm 2^{s+2}\omega\}$.

By Lemma 3, we have

$$\xi^2(\gamma_{l,1})+\xi^2(\gamma_{l,2})=\begin{cases}\pm 2^{n+1}, \text{ if } \mathrm{tr}_1^n\left[\left(1+\overline{\delta_l}\right)^{-1}\right]=0,\\ 0, \quad \text{if } \mathrm{tr}_1^n\left[\left(1+\overline{\delta_l}\right)^{-1}\right]=1.\end{cases} \tag{18}$$

Since $\mathrm{tr}_1^n(0)=\mathrm{tr}_1^n(1)=0$, by the balance property of trace function we have that the equations $\mathrm{tr}_1^n(x)=0$ and $\mathrm{tr}_1^n(x)=1$ have $2^{n-1}-2$ and 2^{n-1} solutions in $\mathbf{F}_{2^n}\setminus\{0,1\}$, respectively.

1) For any fixed $\delta_l\in\mathcal{T}\setminus\{0,1\}$ satisfying $\mathrm{tr}_1^n\left[\left(1+\overline{\delta_l}\right)^{-1}\right]=0$ and for any fixed $0\le j\le 2^n-1$, we discuss the distribution of the pair $(\xi(\gamma_{l,1}),\xi(\gamma_{l,2}))$ when i varies from 0 to 2^n-1.

In the case of $l=1$, by (18) the pair $(\xi(\gamma_{1,1}),\xi(\gamma_{1,2}))$ takes at most 8 values as in Table 6.

By (9), we have $R_{u_i,u_j}(\tau)=2\omega\mathrm{Im}(\xi(\gamma_{1,1})+\xi(\gamma_{1,2}))$. Further, we have

$$\begin{cases}\gamma_{1,1}=1+2\eta_i+2-(1+2\eta_j+2)\delta_1,\\ \gamma_{1,2}=1+2\eta_i+2-(1+2\eta_j)\delta_1.\end{cases} \tag{19}$$

In the case of $l=3$, we similarly have $R_{u_i,u_j}(\tau)=2\omega\mathrm{Im}(\xi(\gamma_{3,2})-\xi(\gamma_{3,1}))$ and

$$\begin{cases}\gamma_{3,1}=1+2\eta_i+2-(1+2\eta_j)\delta_3,\\ \gamma_{3,2}=1+2\eta_i+2-(1+2\eta_j+2)\delta_3.\end{cases} \tag{20}$$

Table 6. The distribution of the pair $(\xi(\gamma_{1,1}), \xi(\gamma_{1,2}))$ when $\mathrm{tr}_1^n\left[\left(1+\overline{\delta_1}\right)^{-1}\right]=0$

$(\xi(\gamma_{1,1}),\xi(\gamma_{1,2}))$	$\mathrm{Im}(\xi(\gamma_{1,1})+\xi(\gamma_{1,2}))$	Frequency
$(2^s,2^s)$	0	x_1
$(2^s,-2^s)$	0	x_2
$(-2^s,2^s)$	0	x_3
$(-2^s,-2^s)$	0	x_4
$(\omega 2^s,\omega 2^s)$	2^{s+1}	x_5
$(\omega 2^s,-\omega 2^s)$	0	x_6
$(-\omega 2^s,\omega 2^s)$	0	x_7
$(-\omega 2^s,-\omega 2^s)$	-2^{s+1}	x_8

Table 7. The distribution of the pair $(\xi(\gamma_{3,2}),\xi(\gamma_{3,1}))$ when $\mathrm{tr}_1^n\left[\left(1+\overline{\delta_3}\right)^{-1}\right]=0$

$(\xi(\gamma_{3,2}),\xi(\gamma_{3,1}))$	$\mathrm{Im}(\xi(\gamma_{3,2})-\xi(\gamma_{3,1}))$	Frequency
$(2^s,2^s)$	0	x_1
$(2^s,-2^s)$	0	x_2
$(-2^s,2^s)$	0	x_3
$(-2^s,-2^s)$	0	x_4
$(\omega 2^s,\omega 2^s)$	0	x_5
$(\omega 2^s,-\omega 2^s)$	2^{s+1}	x_6
$(-\omega 2^s,\omega 2^s)$	-2^{s+1}	x_7
$(-\omega 2^s,-\omega 2^s)$	0	x_8

By (19) and (20), we have that the pair $(\xi(\gamma_{1,1}),\xi(\gamma_{1,2}))$ has the same distribution with $(\xi(\gamma_{3,2}),\xi(\gamma_{3,1}))$, whose distribution is given by Table 7.

By Lemma 2, we have $x_5+x_6=x_7+x_8=2^{n-2}$. And notice that $\sum_{i=1}^{8} x_i=2^n$. Then by Lemma 6, and Tables 6, 7, we have the distribution of the multi-set

$$\left\{R_{u_i,u_j}(\tau): 0\le i,j<2^n, l\in\{1,3\}, \mathrm{tr}_1^n\left[\left(1+\overline{\delta_l}\right)^{-1}\right]=0\right\}$$

as

$$R_{u_i,u_j}(\tau)=\begin{cases}0, & 3\cdot 2^{2n}(2^{n-2}-1) \text{ times },\\ \pm\omega 2^{s+2}, & 2^{2n-1}(2^{n-2}-1) \text{ times (each) }.\end{cases}$$

2) For any fixed $\delta_l\in\mathcal{T}\setminus\{0,1\}$ satisfying $\mathrm{tr}_1^n\left[\left(1+\overline{\delta_l}\right)^{-1}\right]=1$, by (18), we have that $(\xi(\gamma_{l,1}),\xi(\gamma_{l,2}))=(\pm 2^s,\pm\omega 2^s)$ or $(\pm\omega 2^s,\pm 2^s)$. This together with (9) and (17), we can get

$$\begin{aligned}R_{u_i,u_j}(\tau)&=\begin{cases}2\omega\mathrm{Im}(\xi(\gamma_{1,1})+\xi(\gamma_{1,2})), \text{ if } \ l=1 \ ,\\ 2\omega\mathrm{Im}(\xi(\gamma_{3,2})-\xi(\gamma_{3,1})), \text{ if } \ l=3\end{cases}\\ &=\pm\omega 2^{s+1} \ .\end{aligned}$$

Then by Lemma 6, for $l \in \{1,3\}$, we have that $R_{u_i,u_j}(\tau) = \pm\omega 2^{s+1}$ occurs 2^{3n-2} times (each) as i, j range from 0 to 2^n-1, δ_l runs through $\mathcal{T}\backslash\{0,1\}$ and satisfies $\mathrm{tr}_1^n\left[\left(1+\overline{\delta_l}\right)^{-1}\right]=1$.

By the above analysis together with the other four cases, we can get the correlation distribution of the family $\mathcal{S}$ as (16). □

Example 2. For $n=6$, we use the primitive basic irreducible polynomial $f(x)=x^6+3x^5+2x^3+1$ in $\mathbf{Z}_4[x]$ to generate the Galois ring $\mathbf{R}$ =GR$(4,6)$. By computer computation, the family $\mathcal{S}$ defined in Definition 2 can be verified to have the correlation distribution as in Table 8, which is consistent with Theorem 2.

Table 8. The correlation distribution of the family $\mathcal{S}$ for $n=6$

Value	252	-4	0	-36	28	-16ω	-16ω	-32ω	32ω
Frequency	64	131008	450560	47616	79360	131072	131072	30720	30720

5 Conclusion

A new family of quadriphase sequences with low correlation has been constructed, and the correlation distribution of these sequences were determined. The periods of the proposed quadriphase sequences are different from those of previously known quadriphase sequences. This provides more quadriphase sequences with different parameters. For an odd n, the maximum nontrivial correlation magnitude of the proposed family is $4+2^{\frac{n+3}{2}}$, while for an even n, the maximum nontrivial correlation magnitude will be $4+2^{\frac{n+4}{2}}$, which is not better than that of the case for odd n.

Acknowledgement. The authors would like to thank anonymous referees for helpful suggestion on revising this paper. This work of the first two authors was supported by the National Natural Science Foundation of China (NSFC) under Grant 60973130 and the Natural Science Foundation for Excellent Youth Scholars of Hubei Province of China (2009CDA147). The work of the third author was supported by the NSFC (61070172 and 10990011) and the National Basic Research Program of China (2007CB311201).

References

1. Boztas, S., Hammons, R., Kumar, P.V.: 4-phase sequences with near-optimum correlation properties. IEEE Trans. Inf. Theory 38, 1103–1113 (1992)
2. Brown, E.H.: Generalizations of the Kervaire invariant. Ann. Math. 95, 368–383 (1972)
3. Gold, R.: Maximal recursive sequences with 3-valued cross-correlation functions. IEEE Trans. Inf. Theory 14, 154–156 (1968)

4. Jiang, W.F., Hu, L., Tang, X.H., Zeng, X.Y.: New family of binary sequences of period $4(2^n - 1)$ with low correlation. Appl. Algebra Eng. Commun. Comput. 19, 429–439 (2008)
5. Jiang, W.F., Hu, L., Tang, X.H., Zeng, X.Y.: New optimal quadriphase sequences with larger linear span. IEEE Trans. Inf. Theory 55, 458–470 (2009)
6. Kasami, T.: Weight distributions of Bose-Chaudhuri-Hocquenghem codes. In: Bose, R.C., Dowling, T.A. (eds.) Combinatorial Mathematics and Its Applications, pp. 335–357. Univ. North Carolina Press, NC (1969)
7. Kumar, P.V., Helleseth, T., Calderbank, A.R., Hammons Jr., A.R.: Large families of quaternary sequences with low correlation. IEEE Trans. Inf. Theory 42, 579–592 (1996)
8. Li, N., Tang, X.H., Zeng, X.Y., Hu, L.: On the correlation distributions of optimal quaternary sequence family $\mathcal{U}$ and optimal binary sequence family $\mathcal{V}$. IEEE Trans. Inf. Theory (to appear)
9. MacDonald, B.R.: Finite Rings with Identity. Marcel Dekker, Inc., New York (1974)
10. Schmidt, K.-U.: $\mathbf{Z}_4$-valued quadratic forms and quaternary sequences families. IEEE Trans. Inf. Theory 42, 579–592 (1996)
11. Sidelnikov, V.M.: On mutual correlation of sequences. Soviet Math. Dokl 12, 197–201 (1971)
12. Solé, P.: A quaternary cyclic code, and a familly of quadriphase sequences with low correlation properties. In: Wolfmann, J., Cohen, G. (eds.) Coding Theory 1988. LNCS, vol. 388, pp. 193–201. Springer, Heidelberg (1989)
13. Tang, X.H., Udaya, P.: A note on the optimal quadriphase sequences families. IEEE Trans. Inform. Theory 53, 433–436 (2007)
14. Udaya, P., Siddiqi, M.U.: Optimal and suboptimal quadriphase sequences derived from maximal length sequences over $\mathbf{Z}_4$. Appl. Algebra Eng. Commun. Comput. 9, 161–191 (1998)
15. Welch, L.R.: Lower bounds on the maximum crosscorrelation on the signals. IEEE Trans. Inf. Theory 20, 397–399 (1974)
16. Zeng, X.Y., Liu, J.Q., Hu, L.: Generalized Kasami sequences: the large set. IEEE Trans. Inf. Theory 53, 2587–2598 (2007)

On the Link of Some Semi-bent Functions with Kloosterman Sums

Sihem Mesnager[1] and Gérard Cohen[2]

[1] Department of Mathematics, University of Paris VIII and University of Paris XIII, CNRS UMR 7539 LAGA (Laboratoire Analyse, Géometrie et Applications), France
mesnager@math.jussieu.fr
[2] Ecole Nationale Supérieure des Télécommunications -Telecom-Paristech, UMR 5141, CNRS, France
cohen@telecom-paristech.fr

Abstract. We extensively investigate the link between the semi-bentness property of some Boolean functions in polynomial forms and Kloosterman sums.

Keywords: Boolean function, Semi-bent function, Walsh-Hadamard transformation, Kloosterman sums.

1 Introduction

Let n be a positive integer. A Boolean function f on $\mathbb{F}_{2^n}$ is an $\mathbb{F}_2$-valued function over the Galois field $\mathbb{F}_{2^n}$ of order 2^n. Boolean functions play an important role in coding theory (and in particular the class of Reed-Muller codes [20], [9]) on one hand and symmetric cryptography (block ciphers and stream ciphers) [2] on the other hand. Various criteria related to cryptographically desirable Boolean functions have been proposed, such as balancedness, high nonlinearity, correlation immunity, satisfiability of the propagation criterion etc.

The notion of *semi-bent function* has been introduced by Chee, Lee and Kim at Asiacrypt' 94 [7]. In fact, these functions had been previously investigated under the name of three-valued almost optimal Boolean functions in [1]. Moreover, they are particular cases of the so-called plateaued functions [30,29]. Semi-bent functions are widely studied in cryptography because, besides having low Hadamard transform which provides protection against fast correlation attacks [22] and linear cryptanalysis [21], they possess desirable properties such as low autocorrelation, propagation criteria, resiliency and high algebraic degree. Semi-bent functions have been paid a lot of attention in code division multiple access (CDMA) communication systems for sequence design [12], [26], [13], [14], [15], [16], [17] etc. In fact, highly nonlinear functions correspond to sequences that have low cross-correlation with the m-sequences (maximum-length linear feedback shift -register sequences) represented by an absolute trace function $Tr_1^m(x)$. Semi-bent functions exist for even or odd number of variables. When n is even, the semi-bent functions are those Boolean functions whose Hadamard transform

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 263–272, 2011.
© Springer-Verlag Berlin Heidelberg 2011

takes values 0 and $\pm 2^{\frac{n+2}{2}}$. They are balanced (up to the addition of a linear function) and have maximal non-linearity for balanced plateaued functions. When n is odd, the lower bound for the maximum size of the Hadamard transform is not known in general. However, this lower bound has been shown to be $2^{\frac{n+1}{2}}$ when the function is quadratic [20] or when $n = 3, 5, 7$ [25]. Also, it has been shown in [27], [28] that the lower bound for the maximum size of the Hadamard transform does not exceed $\frac{27}{32} \times 2^{\frac{n+1}{2}}$ when $n \geq 15$ is odd. Functions which achieve this lower bound with equality are the semi-bent functions, whose Hadamard transform only takes on the three values 0, $\pm 2^{\frac{n+1}{2}}$ [8].

In this paper, some Boolean functions in polynomial forms with even number of variables are considred. Some papers have been devoted to the construction of semi-bent functions whose expression is a power polynomial $Tr_1^n(x^d)$ for a suitably chosen d and particular values of n (n even). A recent work in this topic is due to Charpin et al. [6] for the construction of quadratic semi-bent functions. A very recent work is [3], in which semi-bent functions in even dimension are characterized and many infinite classes with maximum algebraic degree have been derived from a subclass of bent functions (more precisely, Dillon Partial Spreads $\mathcal{PS}_{ap}$-like). Recall that bent functions are those Boolean functions whose Hadamard transform takes values $\pm 2^{\frac{n}{2}}$. Our main intention in this paper is to investigate the link between Kloosterman sums and the semi-bentness property of functions defined on $\mathbb{F}_{2^n}$ ($n = 2m$) whose polynomial forms are given by (*):

$$Tr_1^n\left(ax^{r(2^m-1)}\right) + Tr_1^2\left(bx^{\frac{2^n-1}{3}}\right) + Tr_1^m\left(c'x^{2^m+1}\right) + Tr_1^n\left(dx^{(2^m-1)s+1}\right)$$

where r is a positive integer such that $gcd(r, 2^m + 1) = 1$, $s \in \left\{0, \frac{1}{4}, \frac{1}{6}, 3\right\}$ ($\frac{1}{4}$ and $\frac{1}{6}$ are understood modulo $2^m + 1$), $a \in \mathbb{F}_{2^m}^\star$, $b \in \mathbb{F}_4$, $c' \in \mathbb{F}_{2^m}$ and $d \in \mathbb{F}_2$. The paper is organized as follows. In section 2, we fix our main notation and recall the necessary background. Next, in section 3, we give some technical results. Finally, in section 4, we characterize some infinite classes of semi-bent parameterized Boolean functions of the form (*) in terms of the evaluation of the classical Kloosterman sum on the single parameter a.

2 Notation and Preliminaries

For any set E, we will denote $E \setminus \{0\}$ by $E^\star$.

• *Boolean functions in polynomial forms*
For any positive integer k, and for any r dividing k, the trace function from $\mathbb{F}_{2^k}$ to $\mathbb{F}_{2^r}$, denoted by Tr_r^k, is the mapping defined as: $\forall x \in \mathbb{F}_{2^k}$, $Tr_r^k(x) := \sum_{i=0}^{\frac{k}{r}-1} x^{2^{ir}}$. In particular, the *absolute trace* over $\mathbb{F}_2$ is the function $Tr_1^n(x) = \sum_{i=0}^{n-1} x^{2^i}$. Recall that, for every integer r dividing k, the trace function Tr_r^k satisfies the transitivity property, that is, $Tr_1^k = Tr_1^r \circ Tr_r^k$.

Every non-zero Boolean function f defined over $\mathbb{F}_{2^n}$ has a (unique) trace expansion of the form:

$$\forall x \in \mathbb{F}_{2^n}, \quad f(x) = \sum_{j \in \Gamma_n} Tr_1^{o(j)}(a_j x^j) + \epsilon(1 + x^{2^n-1})$$

called its polynomial form, where Γ_n is the set of integers obtained by choosing one element in each cyclotomic coset of 2 modulo $2^n - 1$, $o(j)$ is the size of the cyclotomic coset of 2 modulo $2^n - 1$ containing j, $a_j \in \mathbb{F}_{2^{o(j)}}$ and, $\epsilon = wt(f)$ modulo 2 where $wt(f)$ is the *Hamming weight* of the image vector of f, that is, the cardinality of its support $Supp(f) := \{x \in \mathbb{F}_{2^n} \mid f(x) = 1\}$.

The algebraic degree of f is equal to the maximum 2-weight of an exponent j for which $a_j \neq 0$ if $\epsilon = 0$ and to n if $\epsilon = 1$.

• *Walsh transform and semi-bent functions*
Let f be a Boolean function on $\mathbb{F}_{2^n}$. Its *"sign" function* is the integer-valued function $\chi(f) := (-1)^f$. The *W*alsh Hadamard transform of f is the discrete Fourier transform of χ_f, whose value at $\omega \in \mathbb{F}_{2^n}$ is defined as follows:

$$\forall \omega \in \mathbb{F}_{2^n}, \quad \widehat{\chi_f}(\omega) = \sum_{x \in \mathbb{F}_{2^n}} (-1)^{f(x)+Tr_1^n(\omega x)}.$$

Semi-bent functions [7], [8] can be defined as follows:

Definition 1. *For even n, a Boolean function $f : \mathbb{F}_{2^n} \to \mathbb{F}_2$ is said to be semi-bent if $\widehat{\chi_f}(\omega) \in \{0, \pm 2^{\frac{n+2}{2}}\}$, for all $\omega \in \mathbb{F}_{2^n}$. For odd n, a Boolean function $f : \mathbb{F}_{2^n} \to \mathbb{F}_2$ is said to be semi-bent if $\widehat{\chi_f}(\omega) \in \{0, \pm 2^{\frac{n+1}{2}}\}$, for all $\omega \in \mathbb{F}_{2^n}$.*

It is well known (see for instance [2]) that the algebraic degree of a semi-bent Boolean function defined on $\mathbb{F}_{2^n}$ is at most $\frac{n}{2}$.

• *Niho power functions*
Let $n = 2m$ be an even integer. Recall that a positive integer d (always understood modulo $2^n - 1$) is said to be a *Niho exponent*, and x^d is a *Niho power function*, if the restriction of x^d to $\mathbb{F}_{2^m}$ is linear or in other words $d \equiv 2^j \pmod{2^m - 1}$ for some $j < n$. As we consider $Tr_1^n(x^d)$, without loss of generality, we can assume that d is in the normalized form, with $j = 0$, and then we have a unique representation $d = (2^m - 1)s + 1$ with $2 \leq s \leq 2^m$. Four examples of infinite classes of Niho bent functions are known up to linear equivalence. The simplest one is the quadratic function $x \mapsto Tr_1^m\left(cx^{2^m+1}\right)$; $c \in \mathbb{F}_{2^m}^\star$. Three infinite classes of Niho bent functions in univariate form have been given in [11]:

1. $x \mapsto Tr_1^n\left(a_1 x^{(2^m-1)\frac{1}{2}+1} + a_2 x^{(2^m-1)3+1}\right)$, $a_1 \in \mathbb{F}_{2^n}^\star$, $a_2 \in \mathbb{F}_{2^n}^\star$; (if $m \equiv 2 \pmod 4$ then a_2 must be a fifth power of an element in $\mathbb{F}_{2^n}$; otherwise a_2 can be any nonzero element of $\mathbb{F}_{2^n}$).
2. $x \mapsto Tr_1^n\left(a_1 x^{(2^m-1)\frac{1}{2}+1} + a_2 x^{(2^m-1)\frac{1}{4}+1}\right)$, $a_1 \in \mathbb{F}_{2^n}^\star$, $a_2 \in \mathbb{F}_{2^n}^\star$, m odd;
3. $x \mapsto Tr_1^n\left(a_1 x^{(2^m-1)\frac{1}{2}+1} + a_2 x^{(2^m-1)\frac{1}{6}+1}\right)$, $a_1 \in \mathbb{F}_{2^n}^\star$, $a_2 \in \mathbb{F}_{2^n}^\star$ m even.

• *Kloosterman sums*
We need to introduce a classical binary exponential sum on $\mathbb{F}_{2^m}$ (where m is an arbitrary positive integer):

Definition 2. *The classical binary Kloosterman sums on* $\mathbb{F}_{2^m}$ *are:*

$$K_m(a) := \sum_{x \in \mathbb{F}_{2^m}} \chi\big(Tr_1^m(ax + \frac{1}{x})\big), \quad a \in \mathbb{F}_{2^m}$$

The Kloosterman sums are generally defined on the multiplicative group $\mathbb{F}_{2^m}^\star$ of $\mathbb{F}_{2^m}$. In this paper we extend to 0 assuming that $\chi(Tr_1^m(\frac{1}{x})) = 1$ for $x = 0$ (in fact, $Tr_1^m(\frac{1}{x}) = Tr_1^m(x^{2^{m-1}-1})$).

The following Proposition is directly obtained from the result of Lachaud and Wolfmann in [18] which is suitable for any m (even or odd).

Proposition 3. *[18] Let* m *be a positive integer. The set* $\{K_m(a), a \in \mathbb{F}_{2^m}\}$ *is the set of all the integers multiple of* 4 *in the range* $[-2^{(m+2)/2}+1, 2^{(m+2)/2}+1]$.

• *Some additional background*
Let $n = 2m$ be an even integer. Let x be an element of $\mathbb{F}_{2^n}$. The conjugate of x over a subfield $\mathbb{F}_{2^m}$ of $\mathbb{F}_{2^n}$ will be denoted by $\bar{x} = x^{2^m}$ and the relative norm with respect to the quadratic field extension $\mathbb{F}_{2^n}/\mathbb{F}_{2^m}$ by $norm(x) = x\bar{x}$. Also, we denote by U the set $\{u \in \mathbb{F}_{2^n} \mid norm(u) = 1\}$, which is the group of $(2^m + 1)$-st roots of unity. Note that since the multiplicative group of the field $\mathbb{F}_{2^n}$ is cyclic and 2^m+1 divides 2^n-1, the order of U is 2^m+1. Finally, note that the unit 1 is the single element in $\mathbb{F}_{2^m}$ of norm one and every non-zero element x of $\mathbb{F}_{2^n}$ has a unique decomposition as: $x = yu$ with $y \in \mathbb{F}_{2^m}^\star$ and $u \in U$.

3 Some Technical Results

We state a well-known result (different proofs can be found in [18], [10], [19],[5]).

Proposition 4. *Let* $n = 2m$, r *a positive integer such that* $\gcd(r, 2^m + 1) = 1$ *and* $a \in \mathbb{F}_{2^m}$. *Let* U *be the group of* $(2^m + 1)$*-st roots of unity. Then,*

$$\sum_{u \in U} \chi\left(Tr_1^n(au^r)\right) = 1 - K_m(a)$$

The following result can be derived from [24], which extends Proposition 4.

Proposition 5. *Let* $n = 2m$ *with* m *odd and* r *a positive integer such that* $\gcd(r, 2^m + 1) = 1$. *Let* U *be the group of* $(2^m + 1)$*-st roots of unity. Let* $b \in \mathbb{F}_4^\star$, $a \in \mathbb{F}_{2^m}$ *and* ζ *be a generator of the cyclic group* U. *Then,*

$$\sum_{u \in U} \chi\left(Tr_1^n(au^r) + Tr_1^2(bu^{\frac{2^m+1}{3}})\right) = \begin{cases} \frac{K_m(a)-1+4C_m(a,a)}{3} & \text{if } b = 1 \\ \frac{K_m(a)-1-2C_m(a,a)}{3} & \text{if } b \neq 1 \end{cases}$$

Thanks to Proposition 5 one can prove the following result.

Corollary 6. *Let $n = 2m$ with $m > 3$ odd and r be a positive integer such that $\gcd(r, 2^m + 1) = 1$. Let U be the group of $(2^m + 1)$-st roots of unity. Let $a \in \mathbb{F}_{2^m}$ and $b \in \mathbb{F}_4$. Then*

$$\sum_{u \in U} \chi\left(Tr_1^n(au^r) + Tr_1^2(bu^{\frac{2^m+1}{3}})\right) = 1 \tag{1}$$

if and only if

- *$b = 0$ and $K_m(a) = 0$,*
- *or, $b \neq 0$ and $K_m(a) = 4$.*

4 Some Semi-bent Functions in Univariate Forms

In this section, we investigate under which conditions on its coefficients a Boolean function whose univariate form is given by (2) ($\frac{1}{2}$ is understood modulo $2^m + 1$) is semi-bent.

$$\begin{aligned} &Tr_1^n\left(ax^{r(2^m-1)}\right) + Tr_1^2\left(bx^{\frac{2^n-1}{3}}\right) + Tr_1^n\left(cx^{(2^m-1)\frac{1}{2}+1}\right) \\ &\quad + Tr_1^n(dx^{(2^m-1)s+1}) \end{aligned} \tag{2}$$

where r is a positive integer such that $gcd(r, 2^m + 1) = 1$, $s \in \left\{0, \frac{1}{4}, \frac{1}{6}, 3\right\}$ ($\frac{1}{4}$ and $\frac{1}{6}$ are understood modulo $2^m + 1$), $a \in \mathbb{F}_{2^m}^\star$, $b \in \mathbb{F}_4$, $c \in \mathbb{F}_{2^n}$ and $d \in \mathbb{F}_2$. Firstly, we introduce the following decomposition $\mathbb{F}_{2^n}^\star = \bigcup_{u \in U} u\mathbb{F}_{2^m}^\star$. Let $g_{a,b,c,d}^{(r,s)}$ be of any Boolean function of the form (2); note that the restriction of $g_{a,b,c,d}^{(r,s)}$ to any coset $u\mathbb{F}_{2^m}^\star$ ($u \in U$), is affine. More precisely,

- If $b \neq 0$, we consider the functions $g_{a,b,c,d}^{(r,s)}$ of the form (2) only when m is odd. We then have, thanks to the transitivity of the trace function:

$$\forall y \in \mathbb{F}_{2^m}^\star, g_{a,b,c,d}^{(r,s)}(uy) = Tr_1^m(\alpha_u y) + \beta_u \tag{3}$$

  with

$$\begin{aligned} \alpha_u &= Tr_m^n\left(du^{(2^m-1)s+1} + cu^{(2^m-1)\frac{1}{2}+1}\right) = Tr_m^n\left(du^{(2^m-1)s+1} + c\right), \\ \beta_u &= Tr_1^n\left(au^{r(2^m-1)}\right) + Tr_1^2\left(bu^{\frac{2^n-1}{3}}\right). \end{aligned}$$

- Otherwise (that is, $b = 0$), we consider the functions $g_{a,0,c,d}^{(r,s)}$ of the form (2) (without condition on the parity of m). Then, we have thanks to the transitivity of the trace function:

$$\forall y \in \mathbb{F}_{2^m}^\star, g_{a,0,c,d}^{(r,s)}(uy) = Tr_1^m(\alpha_u y) + \beta_u \tag{4}$$

with

$$\alpha_u = Tr_m^n\left(du^{(2^m-1)s+1} + c\right),$$
$$\beta_u = Tr_1^n\left(au^{r(2^m-1)}\right).$$

Therefore, the Walsh transform of a generic function of the form (2) can be computed as follows.

Lemma 7. *Let U be the group of (2^m+1)-st roots of unity. The Walsh transform of a generic element of the form (2) is (taking the same notation as in (3) or (4)) :*

$$\forall \omega \in \mathbb{F}_{2^n},\ \widehat{\chi_{g_{a,b,c,d}^{(r,s)}}}(\omega) = 1 - \sum_{u\in U}\chi(\beta_u) + 2^m \sum_{u\in U}\delta_0(\alpha_u + Tr_m^n(\omega u))\chi(\beta_u) \quad (5)$$

where δ_0 is the indicator of the singleton $\{0\}$, that is, $\delta_0(z) = 1$ if $z = 0$ and 0 otherwise.

Proof. Suppose m odd and $b \neq 0$. Let $\omega \in \mathbb{F}_{2^n}$. The Walsh transform of $g_{a,b,c,d}^{(r,s)}$ is defined as

$$\widehat{\chi_{g_{a,b,c,d}^{(r,s)}}}(\omega) = \sum_{x\in\mathbb{F}_{2^n}} \chi(g_{a,b,c,d}^{(r,s)}(x) + Tr_1^n(wx)).$$

Any element $x \in \mathbb{F}_{2^n}^\star$ having a unique polar decomposition $x = uy$ with $u \in U$ and $y \in \mathbb{F}_{2^m}^\star$, we have:

$$\begin{aligned}\widehat{\chi_{g_{a,b,c,d}^{(r,s)}}}(\omega) &= 1 + \sum_{u\in U}\sum_{y\in\mathbb{F}_{2^m}^\star} \chi(g_{a,b,c,d}^{(r,s)}(uy) + Tr_1^n(wuy))\\ &= 1 + \sum_{u\in U}\sum_{y\in\mathbb{F}_{2^m}^\star} \chi\left(Tr_1^m\left((\alpha_u + Tr_m^n(wu))y\right) + \beta_u\right)\\ &= 1 - \sum_{u\in U}\chi(\beta_u) + 2^m\sum_{u\in U}\delta_0(\alpha_u + Tr_m^n(\omega u))\chi(\beta_u)\end{aligned}$$

because $\sum_{y\in\mathbb{F}_{2^m}}\chi(Tr_1^m(\lambda y)) = 0$ if $\lambda \neq 0$ and 2^m if $\lambda = 0$. Likewise, one can establish (5) by similar calculations when $b = 0$ (for m even or odd).

We are now going to investigate several subfamilies of (2). We begin with a preliminary technical statement.

Lemma 8. *Let $w \in \mathbb{F}_{2^n}^\star$ and $c \in \mathbb{F}_{2^n}^\star \setminus \mathbb{F}_{2^m}$. The number of $u \in U$ such that $Tr_m^n(wu + c) = 0$ equals 0 or 2.*

Proof. One has

$$\begin{aligned}Tr_m^n(wu + c) = 0 &\iff wu + w^{2^m}u^{2^m} + Tr_m^n(c) = 0\\ &\iff u^2 + w^{-1}Tr_m^n(c)u + w^{2^m-1} = 0.\end{aligned}$$

Now recall that the quadratic equation $X^2 + aX + b = 0$, $a \neq 0$, admits 0 or 2 solutions.

Next we recall a result shown in [11].

Lemma 9. *For every $w \in \mathbb{F}_{2^n}$, the three following equations admits 0 or 2 solutions in U.*

$$Tr_m^n(wu + u^{\frac{1}{2}}) = 1 \tag{6}$$

$$Tr_m^n(wu + u^5) = 1 \tag{7}$$

$$Tr_m^n(wu^3 + u^2) + 1 = 0 \tag{8}$$

Now, one can prove the following results.

Theorem 10. *Let $n = 2m$. Let r be a positive integer such that $gcd(r, 2^m+1) = 1$. Let $a \in \mathbb{F}_{2^m}^\star$ and $c \in \mathbb{F}_{2^n}^\star \setminus \mathbb{F}_{2^m}$. Then, $g_{a,0,c,0}^{(r,0)}$ is semi-bent if and only if $K_m(a) = 0$.*

Proof. Using the notation of (4), one has

$$\alpha_u = Tr_m^n(c), \quad \beta_u = Tr_1^n(au^{r(2^m-1)})$$

The equation $Tr_m^n(wu + c) = 0$ admits 0 or 2 solutions for every $w \in \mathbb{F}_{2^n}^\star$ by Lemma 8. Therefore $\sum_{u\in U} \delta_0(\alpha_u + Tr_m^n(wu))\chi(\beta_u) \in \{0, \pm 2\}$ for every $w \in \mathbb{F}_{2^n}^\star$. In the case where $w = 0$, since $\alpha_u = Tr_m^n(c) \neq 0$ because $c \in \mathbb{F}_{2^n} \setminus \mathbb{F}_{2^m}$, $\sum_{u\in U} \delta_0(\alpha_u) = 0$. Basicly, for every $w \in \mathbb{F}_{2^n}$, $\widehat{\chi_{g_{a,0,c,0}^{(r,0)}}}(w) = 1 - \sum_{u\in U} \chi(Tr_1^n(au^{r(2^m-1)}))$ mod 2^{m+1}. Recall that the function $g_{a,0,c,0}^{(r,0)}$ is semi-bent if and only if $\widehat{\chi_{g_{a,0,c,0}^{(r,0)}}}(w) \in \{0, \pm 2^{m+1}\}$ for every $w \in \mathbb{F}_{2^n}$. Now, since

$$-2^{m+1} < -2^m - 1 \leq \sum_{u\in U} \chi(Tr_1^n(au^{r(2^m-1)})) \leq 2^m + 1 < 2^{m+1}$$

then, $g_{a,0,c,0}^{(r,0)}$ is semi-bent if and only if

$$\sum_{u\in U} \chi(Tr_1^n(au^{r(2^m-1)})) = 1.$$

Using the fact that $u \mapsto u^{2^m-1}$ is a permutation of U since $\gcd(2^m-1, 2^m+1) = 1$, we then conclude thanks to Proposition 4.

Remark 11. *The Niho part of a function $g_{a,0,c,0}^{(r,0)}$ in univariate form is $Tr_1^n(cx^{(2^m-1)\frac{1}{2}} + 1)$. Note that*

$$Tr_1^n(cx^{(2^m-1)\frac{1}{2}+1}) = Tr_1^m(Tr_m^n(c^2)x^{2^m+1}).$$

Moreover, recall that $o(j)$ denotes the size of the cyclotomic coset of 2 modulo $2^n - 1$ containing j. We have, $o((2^m-1)\frac{1}{2} + 1) = m$, $o((2^m-1)3 + 1) = n$, $o((2^m-1)\frac{1}{4} + 1) = n$ and $o((2^m-1)\frac{1}{6} + 1) = n$. So in the sequel, it suffices to use the previous identity to get the polynomial form of the presented functions.

Theorem 12. *Let $n = 2m$ with $m > 3$ odd. Let r be a positive integer such that $gcd(r, 2^m + 1) = 1$. Let $a \in \mathbb{F}_{2^m}^\star$, $b \in \mathbb{F}_4^\star$ and $c \in \mathbb{F}_{2^n}^\star \setminus \mathbb{F}_{2^m}$. Then, the function $g_{a,b,c,0}^{(r,0)}$ is semi-bent if and only if $K_m(a) = 4$.*

Proof. Using the notation of (3), we are in the case where

$$\alpha_u = Tr_m^n(c), \quad \beta_u = Tr_1^n(au^{r(2^m-1)}) + Tr_1^2(bu^{\frac{2^n-1}{3}}).$$

Using the same arguments as in the beginning of the proof of Theorem 10, we get that $g_{a,b,c,0}^{(r,0)}$ is semi-bent if and only

$$\sum_{u \in U} \chi(Tr_1^n(au^{r(2^m-1)}) + Tr_1^2(bu^{\frac{2^n-1}{3}})) = 1.$$

We finally conclude thanks to Corollary 6.

Theorem 13. *Let $n = 2m$ with m odd. Let r be a positive integer such that $gcd(r, 2^m+1) = 1$. Let $a \in \mathbb{F}_{2^m}^\star$ and $c \in \mathbb{F}_{2^n}^\star$ such that $Tr_m^n(c) = 1$. Then, $g_{a,0,c,1}^{(r,\frac{1}{4})}$ is semi-bent if and only if, $K_m(a) = 0$.*

Proof. Using the notation of (4), note that we are here in the case where

$$\alpha_u = Tr_n^m(c) + Tr_m^n(u^{(2^m-1)\frac{1}{4}+1}) = 1 + Tr_m^n(u^{\frac{1}{2}})$$
$$\beta_u = Tr_1^n(au^{r(2^m-1)}).$$

Thanks to Lemma 9 (using equation (6)) and noting that $1 + Tr_m^n(u^{\frac{1}{2}})$ has 2 solutions in U, one can repeat the arguments of proof of Theorem 10 and then conclude by Proposition 4.

Thanks to Corollary 6 and Lemma 9 (using equation (6)) one can prove the following result.

Theorem 14. *Let $n = 2m$ with $m > 3$ odd. Let r be a positive integer such that $gcd(r, 2^m + 1) = 1$. Let $a \in \mathbb{F}_{2^m}^\star$, $b \in \mathbb{F}_4^\star$ and $c \in \mathbb{F}_{2^n}^\star$ such that $Tr_m^n(c) = 1$. Then, $g_{a,b,c,1}^{(r,\frac{1}{4})}$ is semi-bent if and only if, $K_m(a) = 4$.*

Thanks to Lemma 9 (using equation (7)) and Proposition 4, one can prove the following result.

Theorem 15. *Let $n = 2m$. Let r be a positive integer such that $gcd(r, 2^m+1) = 1$. Let $a \in \mathbb{F}_{2^m}^\star$ and $c \in \mathbb{F}_{2^n}^\star$ such that $Tr_m^n(c) = 1$. Then, the function $g_{a,0,c,1}^{(r,3)}$ is semi-bent if and only if, $K_m(a) = 0$.*

Thanks to Lemma 9 (using equation (7)) and Corollary 6, one can prove the following result.

Theorem 16. *Let $n = 2m$ with $m > 3$ odd. Let r be a positive integer such that $gcd(r, 2^m + 1) = 1$. Let $a \in \mathbb{F}_{2^m}^\star$, $b \in \mathbb{F}_4^\star$ and $c \in \mathbb{F}_{2^n}^\star$ such that $Tr_m^n(c) = 1$. Then, the function $g_{a,b,c,1}^{(r,3)}$ is semi-bent if and only if, $K_m(a) = 4$.*

Thanks to Lemma 9 (using equation (8)) and Proposition 4, one can prove the following result.

Theorem 17. *Let $n = 2m$ with m even. Let r be a positive integer such that $gcd(r, 2^m + 1) = 1$. Let $a \in \mathbb{F}_{2^m}^\star$ and $c \in \mathbb{F}_{2^n}^\star$ such that $Tr_m^n(c) = 1$. Then, the function $g_{a,0,c,1}^{(r,\frac{1}{6})}$ is semi-bent if and only if, $K_m(a) = 0$.*

Remark 18. *Note that all the functions presented in the previous theorems are of maximal algebraic degree for a semi-bent function, namely m.*

Remark 19. *Note that the characterizations of semi-bent functions given in this paper can also be derived from Theorem 1 in [3] and using the results on bent functions in [24], [23] and [11].*

5 Conclusion

In this paper some functions in polynomial form in even dimension are considered. We derive explicit criteria involving Kloosterman sums for determining whether a function sum of some trace functions, is semi-bent or not. Kloosterman sums are used as a very convenient tool to study the semi-bentness property of those functions.

References

1. Charpin, P., Canteaut, A., Carlet, C., Fontaine, C.: On cryptographic properties of the cosets of R(1,m). IEEE Transactions on Information Theory 47, 1494–1513 (2001)
2. Carlet, C.: Boolean Functions for Cryptography and Error Correcting Codes. In: Crama, Y., Hammer, P.L. (eds.) Chapter of the monography Boolean Models and Methods in Mathematics, Computer Science, and Engineering, pp. 257–397. University Press, Cambridge (2010)
3. Carlet, C., Mesnager, S.: A note on Semi-bent Boolean functions. Cryptology ePrint Archive, Report no 486, `http://eprint.iacr.org/2010/486`
4. Carlitz, L.: Explicit evualation of certain exponential sums. Math. Scand. 44, 5–16 (1979)
5. Charpin, P., Helleseth, T., Zinoviev, V.: The divisibility modulo 24 of Kloosterman sums of $GF(2^m)$, m odd. Journal of Combinatorial Theory, Series A 114, 322–338 (2007)
6. Charpin, P., Pasalic, E., Tavernier, C.: On bent and semi-bent quadratic Boolean functions. IEEE Transactions on Information Theory 51(12), 4286–4298 (2005)
7. Chee, S., Lee, S., Kim, K.: Semi-bent Functions. In: Safavi-Naini, R., Pieprzyk, J.P. (eds.) ASIACRYPT 1994. LNCS, vol. 917, pp. 107–118. Springer, Heidelberg (1995)
8. Cheon, J.H., Chee, S.: Elliptic curves and resilient functions. In: Won, D. (ed.) ICISC 2000. LNCS, vol. 2015, pp. 386–397. Springer, Heidelberg (2001)
9. Cohen, G., Honkala, I., Litsyn, S., Lobstein, A.: Covering codes. North-Holland, Amsterdam (1997)
10. Dillon, J.F., Dobbertin, H.: New cyclic difference sets with Singer parameters. Finite Fields and Their Applications 10(3), 342–389 (2004)

11. Dobbertin, H., Leander, G., Canteaut, A., Carlet, C., Felke, P., Gaborit, P.: Construction of bent functions via Niho Power Functions. Journal of Combinatorial therory, Serie A 113, 779–798 (2006)
12. Gold, R.: Maximal recursive sequences with 3-valued recursive crosscorrelation functions. IEEE Transactions on Information Theory, 14 (1), 154–156 (1968)
13. Helleseth, T.: Some results about the cross-correlation function between two maximal linear sequences. Discr. Math. 16, 209–232 (1976)
14. Helleseth, T.: Correlation of m-sequences and related topics. In: Ding, C., Helleseth, T., Niederreiter, H. (eds.) Proc. SETA 1998. Discrete Mathematics and Theoretical Computer Science, pp. 49–66. Springer, London (1999)
15. Helleseth, T., Kumar, P.V.: Sequences with low correlation. In: Pless, V.S., Huffman, W.C., Brualdi, R.A. (eds.) Handbook of Coding Theory, Part 3: Applications, ch. 21, pp. 1765–1853. Elsevier, Amsterdam (1998)
16. Khoo, K., Gong, G., Stinson, D.R.: A new family of Gold-like sequences. IEEE Trans. Inform. Theory, 181 (2002)
17. Khoo, K., Gong, G., Stinson, D.R.: A new characterization of semibent and bent functions on finite fields. Des. Codes. Cryptogr., 38(2), 279–295 (2006)
18. Lachaud, G., Wolfmann, J.: The weights of the orthogonals of the extended quadratic binary Goppa codes. IEEE Transactions on Information Theory 36(3), 686–692 (1990)
19. Leander, G.: Monomial Bent Functions. IEEE Transactions on Information Theory 2(52), 738–743 (2006)
20. MacWilliams, F.J., Sloane, N.J.: The theory of error-correcting codes. North-Holland, Amsterdam (1977)
21. Matsui, M.: Linear cryptanalysis method for DES cipher. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 386–397. Springer, Heidelberg (1994)
22. Meier, W., Staffelbach, O.: Fast correlation attacks on stream ciphers. In: Günther, C.G. (ed.) EUROCRYPT 1988. LNCS, vol. 330, pp. 301–314. Springer, Heidelberg (1988)
23. Mesnager, S.: Recent Results on Bent and Hyper-bent Functions and Their Link With Some Exponential Sums. In: Proceedings of IEEE Information Theory Workshop, ITW 2010, Dublin (2010)
24. Mesnager, S.: A new class of Bent Boolean functions in polynomial forms. In: Proceedings of International Workshop on Coding and Cryptography, WCC 2009, pp. 5–18 (2009)
25. Mykkeltveit, J.: The covering radius of the (128, 8) reed-muller code is 56. IEEE Transactions on Information Theory 26, 359–362 (1980)
26. Niho, Y.: Multi-valued cross-correlation functions between two maximal linear recursive sequences. Ph.D. dissertation, Univ. Sothern Calif., Los Angeles (1972)
27. Patterson, N.J., Wiedemann, D.H.: The covering radius of the (215, 16) Reed-Muller code is at least 16276. IEEE Transactions on Information Theory 29, 354–356 (1983)
28. Patterson, N.J., Wiedemann, D.H.: Wiedemann. Correction to the covering radius of the (215,16) Reed-Muller code is at least 16276. IEEE Transactions on Information Theory 36, 443 (1990)
29. Zheng, Y., Zhang, X.-M.: Plateaued functions. In: Varadharajan, V., Mu, Y. (eds.) ICICS 1999. LNCS, vol. 1726, pp. 284–300. Springer, Heidelberg (1999)
30. Zheng, Y., Zhang, X.M.: Relationships between bent functions and complementary plateaued functions. In: Song, J.S. (ed.) ICISC 1999. LNCS, vol. 1787, pp. 60–75. Springer, Heidelberg (2000)

Locally Decodable Codes: A Brief Survey

Sergey Yekhanin

Microsoft Research Silicon Valley
yekhanin@microsoft.com

Abstract. Locally decodable codes are error correcting codes that simultaneously provide efficient random-access to encoded data and high noise resilience by allowing reliable reconstruction of an arbitrary data bit from looking at only a small number of randomly chosen codeword bits. Local decodability comes at the price of certain loss in terms of code efficiency. Specifically, locally decodable codes require longer codeword lengths than their classical counterparts. In this work we briefly survey the recent progress in constructions of locally decodable codes.

1 Introduction

Locally Decodable Codes (LDCs) are a special kind of error-correcting codes. Error-correcting codes are used to ensure reliable transmission of information over noisy channels as well as to ensure reliable storage of information on a medium that may be partially corrupted over time (or whose reading device is subject to errors). In both of these applications the message is typically partitioned into small blocks and then each block is encoded separately. Such encoding strategy allows efficient random-access retrieval of the information, since one needs to decode only the portion of data one is interested in. Unfortunately, this strategy yields very poor noise resilience, since in case even a single block (out of possibly tens of thousands) is completely corrupted some information is lost. In view of this limitation it would seem preferable to encode the whole message into a single codeword of an error-correcting code. Such solution clearly improves the robustness to noise, but is also hardly satisfactory, since one now needs to look at the whole codeword in order to recover any particular bit of the message (at least in the case when classical error-correcting codes are used). Such decoding complexity is prohibitive for modern massive data-sets.

Locally decodable codes are error-correcting codes that avoid the problem mentioned above by having extremely efficient *sublinear-time* decoding algorithms. More formally, an r-query locally decodable code C encodes k-symbol messages $\mathbf{x}$ in such a way that one can probabilistically recover any symbol $\mathbf{x}(i)$ of the message by querying only r symbols of the (possibly corrupted) codeword $C(\mathbf{x})$, where r can be as small as 2.

Hadamard code. The classical Hadamard code [MS] encoding k-bit messages to 2^k-bit codewords provides the simplest nontrivial example of locally decodable codes. In what follows, let $[k]$ denote the set $\{1, \ldots, k\}$. Every coordinate in the

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 273–282, 2011.
© Springer-Verlag Berlin Heidelberg 2011

Hadamard code corresponds to one (of 2^k) subsets of $[k]$ and stores the XOR of the corresponding bits of the message $\mathbf{x}$. Let $\mathbf{y}$ be an (adversarially corrupted) encoding of $\mathbf{x}$. Given an index $i \in [k]$ and $\mathbf{y}$, the Hadamard decoder picks a set S in $[k]$ uniformly at random and outputs the XOR of the two coordinates of $\mathbf{y}$ corresponding to sets S and $S \triangle \{i\}$. (Here, $\triangle$ denotes the symmetric difference of sets such as $\{1,4,5\} \triangle \{4\} = \{1,5\}$, and $\{1,4,5\} \triangle \{2\} = \{1,2,4,5\}$). It is not difficult to verify that if $\mathbf{y}$ differs from the correct encoding of $\mathbf{x}$ in at most δ fraction of coordinates then with probability $1 - 2\delta$ both decoder's queries go to uncorrupted locations. In such case, the decoder correctly recovers the i-th bit of $\mathbf{x}$. The Hadamard code allows for a super-fast recovery of the message bits (such as, given a codeword corrupted in 0.1 fraction of coordinates, one is able to recover any bit of the message with probability 0.8 by reading only two codeword bits).

The main parameters of interest in locally decodable codes are the codeword length and the query complexity. The length of the code measures the amount of redundancy that is introduced into the message by the encoder. The query complexity counts the number of bits that need to be read from the (corrupted) codeword in order to recover a single bit of the message. Ideally, one would like to have both of these parameters as small as possible. One however can not minimize the length and the query complexity simultaneously. There is a trade-off. On one end of the spectrum we have LDCs with the codeword length close to the message length, decodable with somewhat large query complexity. Such codes are useful for data storage and transmission. On the other end we have LDCs where the query complexity is a small constant but the codeword length is large compared to the message length. Such codes find applications in complexity theory and cryptography. The true shape of the trade-off between the codeword length and the query complexity of LDCs is not known. Determining it is a major open problem.

Currently there are three known families of LDCs: classical Reed Muller codes [MS], multiplicity codes [KSY11], and matching vector codes [Yek08, Efr09]. In this brief survey we give a high level review of each of these families. We focus on the main ideas underlying the codes and omit many details. In section 3 we review Reed Muller codes. In section 4 we review multiplicity codes, and in section 5 we review matching vector codes. A detailed survey of a large body of work on LDCs (including a detailed treatment of the constructions, lower bounds, and applications) can be found in [Yek10].

2 Preliminaries

We now set up the necessary notation and formally define LDCs.

- $[k] = \{1, \ldots, k\}$;
- $\mathbb{F}_q$ is a finite field of q elements;
- $\mathbb{F}_q^*$ is the multiplicative group of $\mathbb{F}_q$;
- $(\mathbf{x}, \mathbf{y})$ stands for the dot product of vectors $\mathbf{x}$ and $\mathbf{y}$;

- $d(\mathbf{x}, \mathbf{y})$ denotes the Hamming distance between vectors $\mathbf{x}$ and $\mathbf{y}$, i.e., the number of coordinates where $\mathbf{x}$ and $\mathbf{y}$ differ;
- For $\mathbf{w} \in \mathbb{F}_q^n$ and an integer $l \in [n]$, $\mathbf{w}(l)$ denotes the l-th coordinate of $\mathbf{w}$;
- A D-evaluation of a function h defined over a domain D, is a vector of values of h at all points of D;
- With a slight abuse of terminology we often refer to a dimension n of a vector $\mathbf{x} \in \mathbb{F}_q^n$ as its *length*.

A q-ary LDC encoding k-long messages to N-long codewords has three parameters: r, δ, and ϵ. Informally an (r, δ, ϵ)-locally decodable code encodes k-long messages $\mathbf{x}$ to N-long codewords $C(\mathbf{x})$, such that for every $i \in [k]$, the coordinate value $\mathbf{x}_i$ can be recovered with probability $1 - \epsilon$, by a randomized decoding procedure that makes only r queries, even if the codeword $C(x)$ is corrupted in up to δN locations. Formally [KT00, STV99],

Definition 1. *A q-ary code $C : \mathbb{F}_q^k \to \mathbb{F}_q^N$ is said to be (r, δ, ϵ)-locally decodable if there exists a randomized decoding algorithm $\mathcal{A}$ such that*

1. *For all $\mathbf{x} \in \mathbb{F}_q^k$, $i \in [k]$ and all vectors $\mathbf{y} \in \mathbb{F}_q^N$ such that $d(C(\mathbf{x}), \mathbf{y}) \leq \delta N$:*

$$Pr[\mathcal{A}^{\mathbf{y}}(i) = \mathbf{x}(i)] \geq 1 - \epsilon,$$

   where the probability is taken over the random coin tosses of $\mathcal{A}$.
2. *$\mathcal{A}$ makes at most r queries to $\mathbf{y}$.*

An LDC is called *linear* if C is a linear transformation over $\mathbb{F}_q$. A locally decodable code allows to probabilistically decode any coordinate of a message by probing only few coordinates of its corrupted encoding. A stronger property that is sometimes desirable is that of local correctability, allowing to efficiently recover not only coordinates of the message but also all other coordinates of the encoding. We now formally define Locally Correctable Codes (LCCs).

Definition 2. *A code (set) C in the space $\mathbb{F}_q^N$ is (r, δ, ϵ)-locally correctable if there exists a randomized correcting algorithm $\mathcal{A}$ such that*

1. *For all $\mathbf{c} \in C$, $i \in [N]$ and all vectors $\mathbf{y} \in \mathbb{F}_q^N$ such that $d(\mathbf{c}, \mathbf{y}) \leq \delta N$:*

$$Pr[\mathcal{A}^{\mathbf{y}}(i) = \mathbf{c}(i)] \geq 1 - \epsilon,$$

   where the probability is taken over the random coin tosses of $\mathcal{A}$.
2. *$\mathcal{A}$ makes at most r queries to $\mathbf{y}$.*

It is not hard to show [Yek10] that any linear (r, δ, ϵ)-locally correctable code of dimension k in $\mathbb{F}_q^N$ can be turned into a linear (r, δ, ϵ)-locally decodable code encoding k-long q-ary messages to N-long q-ary codewords.

3 Reed Muller Codes

Generalized Reed Muller (RM) codes are the oldest family LDCs. All later families can be seen their generalizations. RM codes are named after their discoverers, Reed and Muller. Muller discovered the codes [Mul54] in the 1950s, and Reed proposed the majority logic decoding [Ree54]. The key idea behind these codes is that of polynomial interpolation. Messages are encoded by complete evaluations of low degree multivariate polynomials over a finite field. Local decodability is achieved through reliance on the rich structure of short local dependencies between such evaluations at multiple points.

A Reed Muller code is specified by three integer parameters. Namely, a prime power (alphabet size) q, number of variables n, and a degree $d < q - 1$. The q-ary code consists of $\mathbb{F}_q^n$-evaluations of all polynomials of total degree at most d in the ring $\mathbb{F}_q[z_1, \ldots, z_n]$. Such code encodes $k = \binom{n+d}{d}$-long messages over $\mathbb{F}_q$ to q^n-long codewords.

We now show that RM codes are LCCs, presenting the simplest local corrector from [BF90, Lip90]. To recover the value of a degree d polynomial $F \in \mathbb{F}_q[z_1, \ldots, z_n]$ at a point $\mathbf{w} \in \mathbb{F}_q^n$ our local corrector shoots a random affine line through $\mathbf{w}$ and then relies on the local dependency between the values of F at some $d + 1$ points along the line.

Proposition 1. *Let n and d be positive integers. Let q be a prime power, $d < q - 1$; then there exists a linear code of dimension $k = \binom{n+d}{d}$ in $\mathbb{F}_q^N$, $N = q^n$, that is $(d+1, \delta, (d+1)\delta)$-locally correctable for all δ.*

Proof. The code consists of $\mathbb{F}_q^n$-evaluations of all polynomials of total degree at most d in the ring $\mathbb{F}_q[z_1, \ldots, z_n]$. The local correction procedure is the following. Given an evaluation of a polynomial F corrupted in up to δ fraction of coordinates and a point $\mathbf{w} \in \mathbb{F}_q^n$ the local corrector picks a vector $\mathbf{v} \in \mathbb{F}_q^n$ uniformly at random and considers a line

$$L = \{\mathbf{w} + \lambda \mathbf{v} \mid \lambda \in \mathbb{F}_q\}$$

through $\mathbf{w}$. Let S be an arbitrary subset of $\mathbb{F}_q^*$, $|S| = d + 1$. The corrector queries coordinates of the evaluation vector corresponding to points $\mathbf{w} + \lambda \mathbf{v}$, $\lambda \in S$ to obtain values $\{e_\lambda\}$. Next, it recovers the unique univariate polynomial h, $\deg h \leq d$, such that $h(\lambda) = e_\lambda$, for all $\lambda \in S$, and outputs $h(0)$.

Note that in case all queries of our corrector go to uncorrupted locations h is the restriction of F to L, and $h(0) = F(\mathbf{w})$. It remains to note that since each individual query of the corrector goes to a uniformly random location, with probability at least $1 - (d + 1)\delta$, it never query a corrupted coordinate.

3.1 Summary of the Parameters

The method behind Reed Muller codes is simple and general. It yields codes for all possible values of query complexity r, i.e., one can set r to be an arbitrary

function of the message length k by specifying an appropriate relation between the number of variables and the degree of polynomials and letting these parameters grow to infinity. Increasing the degree relative to the number of variables yields shorter codes of larger query complexity. We now specify the parameters of Reed Muller locally decodable codes in the two regimes that are of primary interest to applications, namely, the regime of positive rate (i.e., a setting where the ratio of codeword length to message length stays above some constant), and the regime of constant query complexity:

- For every constant $\epsilon > 0$, Reed Muller codes yield $O(k^{\epsilon})$-query LDCs of rate $\epsilon^{\Omega(1/\epsilon)}$. However, for codes of rate above $1/2$, no nontrivial locality is achieved.
- For every constant $r \geq 2$, Reed Muller codes yield r-query LDCs of length $\exp\left(k^{1/(r-1)}\right)$.[1]

4 Multiplicity Codes

Multiplicity codes [KSY11], are the youngest family of LDCs. These codes generalize Reed Muller codes and greatly improve upon them in the regime of high rate. Note that with Reed Muller codes, for the code to have any distance, the degrees of the polynomials need to be smaller than the field size. Multiplicity codes, however, use much higher degree polynomials (and thus have significantly improved rates), and compensate for the loss in distance by evaluating polynomials together with their *partial derivatives*.

In section 3.1 we noted that Reed Muller based LDCs cannot have rate above $1/2$. One can however get $O\left(\sqrt{k}\right)$-query LDCs of rate arbitrary close to half by setting $n = 2$, and $d = (1-\tau)q$ in proposition 1 and letting q grow to infinity.

In what follows, rather than treating multiplicity codes in full generality, we present the most basic family of such codes that have query complexity $O\left(\sqrt{k}\right)$ and rate close to $2/3$. Later, we explain how general multiplicity codes are defined.

Proposition 2. *Let q be a prime power, $\tau > 0$, and $d \leq 2(1-\tau)(q-1) - 2$ be an integer; then there exists a linear code of dimension $k = \binom{d+2}{2}$ in $\mathbb{F}_q^N$, $N = 3q^2$, that is $(2(q-1), \delta, 12\delta/\tau + 2/q)$-locally correctable for all δ.*

Proof. Codewords of the multiplicity code correspond to polynomials F in the ring $\mathbb{F}_q[z_1, z_2]$ of total degree up to d. Coordinates are organized in triples indexed by elements of $\mathbf{w} \in \mathbb{F}_q^2$. A triple corresponding to a point $\mathbf{w}$ stores the values

$$F(\mathbf{w}), \left.\frac{\partial F}{\partial z_1}\right|_{\mathbf{w}}, \left.\frac{\partial F}{\partial z_2}\right|_{\mathbf{w}}. \tag{1}$$

We omit a simple proof [KSY11] that distinct polynomials F yield distinct codewords. Given a δ-corrupted codeword corresponding to a polynomial F and a point $\mathbf{w} \in \mathbb{F}_q^2$ the local corrector needs to recover the triple (1).

[1] Throughout the paper we use the standard notation $\exp(x) = 2^{O(x)}$.

1. The corrector starts by picking a vector $\mathbf{v}_1 \in \mathbb{F}_q^2$ uniformly at random and considering a line
$$L_1 = \{\mathbf{w} + \lambda \mathbf{v}_1 \mid \lambda \in \mathbb{F}_q\}$$
through $\mathbf{w}$. The goal of the corrector here is to recover the univariate restriction $f_1(\lambda) = F(\mathbf{w} + \lambda \mathbf{v}_1) \in \mathbb{F}_q[\lambda]$. To this end the corrector queries $3(q-1)$ codeword coordinates corresponding to points $\{\mathbf{w} + \lambda \mathbf{v}_1\}_{\lambda \neq 0}$, to obtain the (possibly corrupted) values of F and partial derivatives of F. The corrector then uses these values to recover the (possibly corrupted) values $\{\mathrm{val}_\lambda, \mathrm{der}_\lambda\}_{\lambda \neq 0}$ of f_1 and the derivative of f_1 via the chain rule
$$f_1'(\lambda) = \left.\frac{\partial F}{\partial z_1}\right|_{\mathbf{w}+\lambda\mathbf{v}_1} \mathbf{v}_1(1) + \left.\frac{\partial F}{\partial z_2}\right|_{\mathbf{w}+\lambda\mathbf{v}_1} \mathbf{v}_1(2). \quad (2)$$
Next, the corrector recovers the unique univariate polynomial f_1, $\deg f_1 \leq d$, such that
$$f_1(\lambda) = \mathrm{val}_\lambda \quad \text{and} \quad f_1'(\lambda) = \mathrm{der}_\lambda,$$
for all but at most $\lfloor \tau(q-1)/2 \rfloor$ values of $\lambda \in \mathbb{F}_q^*$. The uniqueness of f_1 if it exists follows from the fact that a degree d nonzero univariate polynomial cannot vanish together with its derivative at more than $d/2$ points. If a polynomial f_1 does not exist; the corrector halts with an arbitrary output. Note that since each individual query of the corrector goes to a uniformly random location, by Markov's inequality the probability that $\tau(q-1)/2$ or more of the queries go to corrupted locations is at most $6\delta/\tau$. Therefore with probability at least $1 - 6\delta/\tau$, the recovered polynomial f_1 is indeed the restriction of F to the line L_1. Thus $f_1(0) = F(\mathbf{w})$, and $f_1'(0)$ is the derivative of F in direction $\mathbf{v}_1$.
2. It is not hard to see that knowing the polynomial f_1 is not sufficient to recover (1). Thus on the second step the corrector again picks a uniformly random vector $\mathbf{v}_2 \in \mathbb{F}_q^2$, considers the line L_2 through $\mathbf{w}$ in direction $\mathbf{v}_2$, and recovers the restriction f_2 of F to line L_2, to obtain the value of the directional derivative $f_2'(0)$ of F in direction $\mathbf{v}_2$.
3. Finally, on the last step, the corrector combines directional derivatives of F in directions $\mathbf{v}_1$ and $\mathbf{v}_2$ to recover the partial derivatives of F at $\mathbf{w}$. It is not hard to show [KSY11] that such a recovery is always possible whenever $\mathbf{v}_1$ and $\mathbf{v}_2$ are not collinear, which happens with probability at least $1 - 2/q$.

Proposition 2 yields $O\left(\sqrt{k}\right)$-query codes of rate arbitrarily close to $2/3$ by evaluating bivariate polynomials together with their first partial derivatives. General multiplicity codes are obtained by evaluating n-variate polynomials together with all their mixed partial derivatives of order up to s, for arbitrary positive integers n and s. Increasing n reduces the query complexity; increasing s yields codes of larger rate.

4.1 Summary of the Parameters

Setting the number of variables, and the order of derivatives appropriately one can for arbitrary constants $\alpha, \epsilon > 0$ get multiplicity codes of rate $1 - \alpha$ and

query complexity $O(k^{\epsilon})$. Multiplicity codes also have respectable concrete parameters [KSY11], and thus are potentially useful in practice.

5 Matching Vector Codes

Matching Vector (MV) locally decodable codes were developed in a sequence of works [Yek08, Rag07, Efr09, IS10, DGY10, BET10a, MFL+10, BET10b, SY11]. In the setting of a constant number of queries these codes have dramatically better parameters than Reed Muller based LDCs. Our presentation of matching vector codes follows the "polynomial-centric" view developed in [DGY10].

An MV code consists of a linear subspace of polynomials in $\mathbb{F}_q[z_1, \ldots, z_n]$, evaluated at all points of $\mathbb{C}_m^n$, where $\mathbb{C}_m$ is a certain multiplicative subgroup of $\mathbb{F}_q^*$. The decoding algorithm is similar to traditional local decoders for RM codes given by proposition 1. The decoder shoots a line in a certain direction and decodes along it. The difference is that the monomials which are used are not of low-degree, they are chosen according to a matching family of vectors. Further, the lines for decoding are *multiplicative*, a notion that we define shortly. In what follows let $\mathbb{Z}_m$ denote the ring of integers modulo an integer m.

Definition 3. *Let $S \subseteq \mathbb{Z}_m \setminus \{0\}$. We say that families $\mathcal{U} = \{\mathbf{u}_1, \ldots, \mathbf{u}_k\}$ and $\mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ of vectors in $\mathbb{Z}_m^n$ form an S-matching family if the following two conditions are satisfied:*

- *For all $i \in [k]$, $(\mathbf{u}_i, \mathbf{v}_i) = 0$;*
- *For all $i, j \in [k]$ such that $i \neq j$, $(\mathbf{u}_j, \mathbf{v}_i) \in S$.*

We now show how one can obtain an MV code out of a matching family. We start with some notation.

- We assume that q is a prime power, m divides $q-1$, and denote a subgroup of $\mathbb{F}_q^*$ of order m by $\mathbb{C}_m$;
- We fix some generator g of $\mathbb{C}_m$;
- For $\mathbf{w} \in \mathbb{Z}_m^n$, we define $g^{\mathbf{w}} \in \mathbb{C}_m^n$ by $\left(g^{\mathbf{w}(1)}, \ldots, g^{\mathbf{w}(n)}\right)$;
- For $\mathbf{w}, \mathbf{v} \in \mathbb{Z}_m^n$ we define the multiplicative line $M_{\mathbf{w},\mathbf{v}}$ through $\mathbf{w}$ in direction $\mathbf{v}$ to be the multi-set

$$M_{\mathbf{w},\mathbf{v}} = \left\{ g^{\mathbf{w}+\lambda\mathbf{v}} \mid \lambda \in \mathbb{Z}_m \right\}; \tag{3}$$

- For $\mathbf{u} \in \mathbb{Z}_m^n$, we define the monomial $\mathrm{mon}_{\mathbf{u}} \in \mathbb{F}_q[z_1, \ldots, z_n]$ by

$$\mathrm{mon}_{\mathbf{u}}(z_1, \ldots, z_n) = \prod_{\ell \in [n]} z_\ell^{\mathbf{u}(\ell)}. \tag{4}$$

We now outline the encoding/decoding framework for matching vector codes. Observe that for any $\mathbf{w}, \mathbf{u}, \mathbf{v} \in \mathbb{Z}_m^n$ and $\lambda \in \mathbb{Z}_m$ we have

$$\mathrm{mon}_{\mathbf{u}}\left(g^{\mathbf{w}+\lambda\mathbf{v}}\right) = g^{(\mathbf{u},\mathbf{w})}\left(g^{\lambda}\right)^{(\mathbf{u},\mathbf{v})}. \tag{5}$$

The formula above implies that the $M_{\mathbf{w},\mathbf{v}}$-evaluation of a monomial $\mathrm{mon}_{\mathbf{u}}$ is a $\mathbb{C}_m$-evaluation of a (univariate) monomial

$$g^{(\mathbf{u},\mathbf{w})}y^{(\mathbf{u},\mathbf{v})} \in \mathbb{F}_q[y]. \tag{6}$$

This observation is the foundation of our local decoder. We now sketch encoding and decoding procedures. Let $\mathcal{U}, \mathcal{V}$ be an S-matching family in $\mathbb{Z}_m^n$.

Encoding: We encode a message $(\mathbf{x}(1), \ldots, \mathbf{x}(k)) \in \mathbb{F}_q^k$ by the $\mathbb{C}_m^n$-evaluation of the polynomial

$$F(z_1, \ldots, z_n) = \sum_{j=1}^{k} \mathbf{x}(j) \cdot \mathrm{mon}_{\mathbf{u}_\mathrm{j}}(\mathrm{z}_1, \ldots, \mathrm{z}_\mathrm{n}). \tag{7}$$

Decoding: The input to the decoder is a (corrupted) $\mathbb{C}_m^n$-evaluation of F and an index $i \in [k]$.

1. The decoder picks $\mathbf{w} \in \mathbb{Z}_m^n$ uniformly at random;
2. The decoder recovers the noiseless restriction of F to $M_{\mathbf{w},\mathbf{v}_i}$. To accomplish this the decoder queries the (corrupted) $M_{\mathbf{w},\mathbf{v}_i}$-evaluation of F at a certain number of locations.

To see that noiseless $M_{\mathbf{w},\mathbf{v}_i}$-evaluation of F uniquely determines $\mathbf{x}(i)$ note that by formulas (5), (6) and (7) the $M_{\mathbf{w},\mathbf{v}_i}$-evaluation of F is a $\mathbb{C}_m$-evaluation of a polynomial

$$f(y) = \sum_{j=1}^{k} \mathbf{x}(j) \cdot g^{(\mathbf{u}_j,\mathbf{w})}y^{(\mathbf{u}_j,\mathbf{v}_i)} \in \mathbb{F}_q[y]. \tag{8}$$

Further observe that properties of the S-matching family $\mathcal{U}, \mathcal{V}$ and (8) yield

$$f(y) = \mathbf{x}(i) \cdot g^{(\mathbf{u}_i,\mathbf{w})} + \sum_{s \in S} \left(\sum_{j \,:\, (\mathbf{u}_j,\mathbf{v}_i)=s} \mathbf{x}(j) \cdot g^{(\mathbf{u}_j,\mathbf{w})} \right) y^s. \tag{9}$$

It is evident from the above formula that the restriction of F to a multiplicative line $M_{\mathbf{w},\mathbf{v}_i}$ yields a univariate polynomial $f(y)$ such that the set of monomial degrees of f is in $S \cup \{0\}$ and

$$\mathbf{x}(i) = f(0)/g^{(\mathbf{u}_i,\mathbf{w})}. \tag{10}$$

Proposition 3. *Let $\mathcal{U}, \mathcal{V}$ be a family of S-matching vectors in $\mathbb{Z}_m^n$, $|\mathcal{U}| = |\mathcal{V}| = k$, $|S| = s$. Suppose $m \mid q-1$, where q is a prime power; then there exists a q-ary linear code encoding k-long messages to m^n-long codewords that is $(s+1, \delta, (s+1)\delta)$-locally decodable for all δ.*

Proof. The encoding procedure has already been specified by formula (7). To recover the value $\mathbf{x}(i)$

1. The decoder picks $\mathbf{w} \in \mathbb{Z}_m^n$ at random, and queries the (corrupted) $M_{\mathbf{w},\mathbf{v}_i}$-evaluation of F at $(s+1)$ consecutive locations $\left\{g^{\mathbf{w}+\lambda\mathbf{v}_i} \mid \lambda \in \{0,\ldots,s\}\right\}$ to obtain values $c_0,\ldots,c_s$.
2. The decoder recovers the unique sparse univariate polynomial $h(y) \in \mathbb{F}_q[y]$ with $\mathrm{supp(h)} \subseteq S \cup \{0\}$ such that for all $\lambda \in \{0,\ldots,s\}$, $h(g^\lambda) = c_\lambda$. (The uniqueness of $h(y)$ follows from standard properties of Vandermonde matrices. [LN83])
3. Following the formula (10) the decoder returns $h(0)/g^{(\mathbf{u}_i,\mathbf{w})}$.

The discussion above implies that if all $(s+1)$ locations queried by the decoder are not corrupted then $h(y)$ is indeed the noiseless restriction of F to $M_{\mathbf{w},\mathbf{v}_i}$, and decoder's output is correct. It remains to note that each individual query of the decoder goes to a uniformly random location and apply the union bound.

5.1 Summary of the Parameters

Parameters of MV codes are determined by parameters of the underlying family of matching vectors. The largest currently know such families are based on Grolmusz's set systems with restricted intersections modulo composites [Gro00]. Plugging the parameters of these families into proposition 3 one gets 2^t-query LDCs of length $\exp\exp\left((\log k)^{1/t}(\log\log k)^{1-1/t}\right)$ for any constant $t \geq 2$. Further modest reductions in query complexity are possible [Efr09, IS10, MFL+10].

6 Conclusions

In this paper we have briefly surveyed the three main families of locally decodable codes. Namely, classical Reed Muller codes; multiplicity codes that are the best LDCs in the regime of high rate; and matching vector codes that are the best LDCs in the regime of low query complexity. We focused on the key ideas underlying the constructions and omitted many details, such as alphabet reduction techniques, and extensions that allow to correct more errors. A longer survey of a large body of work on locally decodable codes (including a detailed treatment of the constructions, lower bounds, and applications) can be found in [Yek10].

References

[BET10a] Ben-Aroya, A., Efremenko, K., Ta-Shma, A.: Local list decoding with a constant number of queries. In: 51st IEEE Symposium on Foundations of Computer Science (FOCS), pp. 715–722 (2010)

[BET10b] Ben-Aroya, A., Efremenko, K., Ta-Shma, A.: A note on amplifying the error-tolerance of locally decodable codes. In: Electronic Colloquium on Computational Complexity (ECCC), TR10-134 (2010)

[BF90] Beaver, D., Feigenbaum, J.: Hiding instances in multioracle queries. In: Choffrut, C., Lengauer, T. (eds.) STACS 1990. LNCS, vol. 415, pp. 37–48. Springer, Heidelberg (1990)

[DGY10] Dvir, Z., Gopalan, P., Yekhanin, S.: Matching vector codes. In: 51st IEEE Symposium on Foundations of Computer Science (FOCS), pp. 705–714 (2010)

[Efr09] Efremenko, K.: 3-query locally decodable codes of subexponential length. In: 41st ACM Symposium on Theory of Computing (STOC), pp. 39–44 (2009)

[Gro00] Grolmusz, V.: Superpolynomial size set-systems with restricted intersections mod 6 and explicit Ramsey graphs. Combinatorica 20, 71–86 (2000)

[IS10] Itoh, T., Suzuki, Y.: New constructions for query-efficient locally decodable codes of subexponential length. IEICE Transactions on Information and Systems, 263–270 (2010)

[KSY11] Kopparty, S., Saraf, S., Yekhanin, S.: High-rate codes with sublinear-time decoding. In: 43nd ACM Symposium on Theory of Computing, STOC (2011)

[KT00] Katz, J., Trevisan, L.: On the efficiency of local decoding procedures for error-correcting codes. In: 32nd ACM Symposium on Theory of Computing (STOC), pp. 80–86 (2000)

[Lip90] Lipton, R.: Efficient checking of computations. In: Choffrut, C., Lengauer, T. (eds.) STACS 1990. LNCS, vol. 415, pp. 207–215. Springer, Heidelberg (1990)

[LN83] Lidl, R., Niederreiter, H.: Finite Fields. Cambridge University Press, Cambridge (1983)

[MFL+10] Chee, Y.M., Feng, T., Ling, S., Wang, H., Zhang, L.: Query-efficient locally decodable codes of subexponential length. In: Electronic Colloquium on Computational Complexity (ECCC), TR10-173 (2010)

[MS] MacWilliams, F.J., Sloane, N.J.A.: The Theory of Error Correcting Codes. North-Holland, Amsterdam

[Mul54] Muller, D.E.: Application of boolean algebra to switching circuit design and to error detection. IEEE Transactions on Computers 3, 6–12 (1954)

[Rag07] Raghavendra, P.: A note on Yekhanin's locally decodable codes. In: Electronic Colloquium on Computational Complexity (ECCC), TR07-016 (2007)

[Ree54] Reed, I.S.: A class of multiple-error-correcting codes and the decoding scheme. IEEE Transactions on Information Theory 4, 38–49 (1954)

[STV99] Sudan, M., Trevisan, L., Vadhan, S.: Pseudorandom generators without the XOR lemma. In: 39th ACM Symposium on Theory of Computing (STOC), pp. 537–546 (1999)

[SY11] Saraf, S., Yekhanin, S.: Noisy interpolation of sparse polynomials, and applications. In: 29th IEEE Computational Complexity Conference, CCC (2011)

[Yek08] Yekhanin, S.: Towards 3-query locally decodable codes of subexponential length. Journal of the ACM 55, 1–16 (2008)

[Yek10] Yekhanin, S.: Locally decodable codes. Foundations and trends in Theoretical Computer Science (2010, to appear)

On Relationship of Computational Diffie-Hellman Problem and Computational Square-Root Exponent Problem*

Fangguo Zhang and Ping Wang

School of Information Science and Technology,
Sun Yat-Sen University, Guangzhou 510006, China
isszhfg@mail.sysu.edu.cn

Abstract. The Computational Square-Root Exponent Problem (CSREP), which is a problem to compute a value whose discrete logarithm is a square root of the discrete logarithm of a given value, was proposed in the literature to show the reduction between the discrete logarithm problem and the factoring problem. The CSREP was also used to construct certain cryptography systems. In this paper, we analyze the complexity of the CSREP, and show that under proper conditions the CSREP is polynomial-time equivalent to the Computational Diffie-Hellman Problem (CDHP). We also demonstrate that in group G with certain prime order p, the DLP, CDHP and CSREP may be polynomial time equivalent with respect to the computational reduction for the first time in the literature.

Keywords: Diffie-Hellman problem, square Diffie-Hellman problem, square-root exponent problem, equivalence.

1 Introduction

Most modern cryptographic systems rely on assumptions on the computational difficulty of some particular number-theoretic problems. Such as the first instance of public key cryptography, the Diffie-Hellman key agreement protocol [3], was designed from the discrete logarithm problem (DLP). Since then, numerous cryptographic systems and protocols have been proposed based their security on the difficulty of solving DLP, such as the ElGamal signature and encryption schemes [4], the U.S. governments Digital Signature Algorithm (DSA) [5], the Schnorr signature scheme [18], etc. In fact, the security of the Diffie-Hellman key agreement protocol rely on the hardness of computational Diffie-Hellman problem (CDHP), not the DLP. Actually, there are not too many cryptosystems that are provable secure under DLP assumption, even they are constructed from DLP.

* This work is supported by the National Natural Science Foundation of China (No. 61070168).

Y.M. Chee et al. (Eds.): IWCC 2011, LNCS 6639, pp. 283–293, 2011.
© Springer-Verlag Berlin Heidelberg 2011

We recall the following two definitions:

Definition 1. (Discrete Logarithm Problem, DLP) *Let G be a cyclic group of order q and let $g \in G$ be generator of G. Given $g, h \in G$ as input, output unique integer $0 \leq a < q$ such that $h = g^a$.*

Definition 2. (Computational Diffie-Hellman Problem, CDHP) *Let G be a cyclic group of order q and let $g \in G$ be generator of G. Given g, g^a, g^b as input, output g^{ab}.*

The CDHP is believed to be closely related to the difficulty of computing DLP in a cyclic group G [12]. According to the Diffie-Hellman key agreement protocol, the values g^a and g^b are publically available. The Diffi-Hellman assumption states that it is computationally difficult to compute g^{ab} form the known values g^a and g^b. It is obvious that if DLP can be easily computed, then one can simply compute a from g^a and then compute $g^{ab} = (g^b)^a$.

Maurer and Wolf [8,9,10,11] have proved that, for every cyclic group G with prime order p, there exists polynomial time algorithm that reduces the computation of DLP in G to the computation of CDHP in G if we are able to find an elliptic curve, called *auxiliary elliptic curve*, over $\mathbb{F}_p$ with smooth order. Moreover, Muzereau *et al.* [13] showed that such auxiliary elliptic curves are highly likely to exist for almost all elliptic curve groups. They built auxiliary elliptic curves with smooth orders for most of the curves in the SECG standard, which means Maurer's proof is applicable to most of the groups used in practical elliptic curve cryptography. This reduction implies that the DLP and CDHP are polynomial time equivalent in G.

Various computational and decisional problems related to the Diffie-Hellman problem have been proposed and analyzed in the literature. The square computational Diffie-Hellman problem (Squ-CDHP), which is a problem to compute an element whose discrete logarithm is a square of the discrete logarithm of a given value, has been studied by a set of researchers [2,9]. Pfitzmann and Sadeghi [15] first proposed using the inverse computational Diffie-Hellman problem (Inv-CDHP) in their anonymous fingerprinting scheme.

Konoma *et al.* [6] defined a new problem called the computational square-root exponent problem (CSREP), which is a problem to compute a value whose discrete logarithm is a square root of the discrete logarithm of a given value, to analyze reduction between the discrete logarithm problem modulo a prime and the factoring problem. They showed that CSREP is the first problem known to stay between the computational Diffie-Hellman problem and the decisional Diffie-Hellman problem with respect to the computational reduction. Furthermore, Zhang *et al.* [19] designed a new signature scheme without random oracles from bilinear pairings and the CSREP. Zhang *et al.* [20] also proposed a new designated confirmer signature scheme from bilinear pairings and the hardness of the CSREP.

The relationship among Squ-CDHP, Inv-CDHP and CDHP have been studied and analyzed in [1] and [17], and there are various cryptographic applications based their security on those problems. In this paper, we analyze the complexity

of the CSREP, and prove that under certain conditions the CSREP is polynomial time equivalent to the CDHP. According to Maurer and Wolf's work [8,9,10,11], we also provided the witness that in group G with certain prime order p, the DLP, CDHP and CSREP may be polynomial time equivalent with respect to the computational reduction for the first time in the literature.

The remainder of this paper is organized as follows. In Section 2, we define certain notations and recall the computational square-root exponent problem. We provide and prove our main result in section 3 and discuss several corollaries. We present an algorithm and certain example in section 4 and conclude the paper in section 5.

2 Some Notations and the Computational Square-Root Exponent Problem

To analyze and clarify the complexity of various cryptography primitives, a useful complexity analysis is to show reductions among these primitives. Generally, to prove the equivalence of two problems, it may be easier to show the reduction relationship between them. Therefore, before describing the mathematical problems, we need the following notations from complexity theory.

- An algorithm is said to be of polynomial time if its running time is upper bounded by a polynomial expression in the size of the input for the algorithm. Here time is identified with steps in computation.
- We say problem **A** is polynomial time reducible to problem **B**, denoted by $\mathbf{B} \Rightarrow \mathbf{A}$, if there exists a polynomial time algorithm $\mathcal{R}$ for solving problem **A** that makes calls to a subroutine for problem **B**. In this case, we also say the problem **B** is *harder* than the problem **A**.
- We say that **A** and **B** are polynomial time equivalent if **A** is polynomial time reducible to **B** and **B** is polynomial time reducible to **A**.

In [6], Konoma *et al.* defined two new problems called Computational Square-Root Exponent Problem (CSREP) and Decisional Square-Root Exponent Problem (DSREP). Konoma *et al.* also introduced a new related problem called Square Root of Discrete Logarithm Problem (SRDLP) in [7]. CSREP is closely related to CDHP and SRDLP is closely related to DLP.

Konoma *et al.*'s definitions for CSREP and DSREP are over the multiplicative group modulo a prime p. We recall the definitions for CSREP and DSREP in any cyclic group with order q as follows:

Definition 3 (CSREP). *Let G be a cyclic group of order q and let $g \in G$ be generator of G. Given g and g^a as input, output $g^{a^{\frac{1}{2}}}$ if a is a quadratic residue modulo q. Otherwise, output $\perp$.*

Definition 4 (DSREP). *Let G be a cyclic group of order q and let $g \in G$ be generator of G. Given g, g^a and y as input, decide whether the discrete logarithm of y is a square root of the discrete logarithm of g^a. That is, output 1 if $y = g^{a^{\frac{1}{2}}}$ and 0 if $y \neq g^{a^{\frac{1}{2}}}$.*

Konoma *et al.*[6] showed the reduction between the discrete logarithm problem modulo a prime and the factoring problem through the square-root exponent problem. They also analyzed the reduction among the CSREP and the DSREP and the Diffie-Hellman problem and show that under some conditions the gap between the complexity of the CSREP and that of the DSREP partially overlaps with the gap between the complexity of the computational Diffie-Hellman problem and that of the decisional Diffie-Hellman problem. Which means CSREP is the first problem known to stay between the computational Diffie-Hellman problem and the decisional Diffie-Hellman problem with respect to the computational reduction. However, we will show later that under proper conditions the CSREP is polynomial-time equivalent to the Computational Diffie-Hellman Problem. Therefore, it seems that the CSREP is not the problem to stay between the CDHP and the DDHP.

Notice that when the order q of G is not a prime, *e.g.*, a RSA module (*i.e.*, it is the product of two safe primes), CSREP may be harder than DLP. This is because that even DLP can be solved, it seems that we still can not solve CSREP due to the computation of the quadratic residue modulo a RSA module. Because of the method of Pohlig and Hellman [16], the hardness of DLP on the group G of order q can be reduced to DLP on the subgroup with the order of the largest prime factor of q. Therefore, for the following sections we focus on the cases where the order of group G is a prime p.

Our reduction will use a variation of CDHP. In fact, there are two variations of CDHP:

- **Inverse Computational Diffie-Hellman Problem (Inv-CDHP):** For $a \in \mathbb{Z}_q^*$, given g, g^a, to compute $g^{a^{-1}}$.
- **Square Computational Diffie-Hellman Problem (Squ-CDHP):** For $a \in \mathbb{Z}_q^*$, given g, g^a, to compute g^{a^2}.

Due to the results of [1,8,17], we have the following theorem:

Theorem 1. *CDHP, Inv-CDHP and Squ-CDHP are polynomial time equivalent.*

3 Relation between CSREP and CDHP

The computing of CSREP requires taking a square root of an exponent without obtaining the exponent itself. In this section, we discuss the computational equivalence between the CSREP and CDHP. More precisely, we have the following main result:

Theorem 2. *Let G be a cyclic group of prime order p and let $g \in G$ be generator of G. Let $p = 4k - 1$, for some $k, i, j \in \mathbb{Z}$, and i, j are polynomial in $\log p$, if one of the following conditions is satisfied:*

1). $k^i \equiv \pm 2^j \mod p - 1$
2). $k \equiv \pm(2^j + 1) \mod p - 1$

3). $k \equiv \pm(2^j - 2^{j-1} + 1) \mod p-1$
4). $k \equiv \pm(2^{2j} \pm 2^j + 1) \mod p-1$
5). $k \equiv \pm(2^{2j+1} \pm 2^j + 1) \mod p-1$

then computing CSREP in G is polynomial time equivalent to computing CDHP in G.

Proof. First, we consider CDHP $\Rightarrow$ CSREP as follows:

According to Theorem 1, computing CDHP in G is polynomial time equivalent to computing Squ-CDHP in G. We assume that there are oracles A_1 and A_2, which can compute CDHP and Squ-CDHP, respectively. Which means, on input g, g^a and g^b for $a, b \in \mathbb{Z}_p^*$, we can compute $g^{ab} = A_1(g, g^a, g^b)$, and on input g, g^a for $a \in \mathbb{Z}_p^*$, we can compute $g^{a^2} = A_2(g, g^a)$.

Now given g and g^a, we want to compute $g^{a^{\frac{1}{2}}}$, when a is a quadratic residue modulo p. Note that, for any $i \in \mathbb{Z}$, we can compute g^{a^i} by using oracle A_1 and A_2. On the other hand, when a is a quadratic residue modulo p, $(a^{\frac{1}{2}} \mod p)$ is always a polynomial $f(a)$ of a, then $g^{a^{\frac{1}{2}}} = g^{f(a)}$. Which means we can always compute $g^{a^{\frac{1}{2}}}$ by using oracle A_1 and A_2. Moreover, we can compute $a^{\frac{1}{2}}$ efficiently by using the algorithm of [14] in polynomial time. Thus, we can solve CSREP in G using CDHP oracle.

Then, we prove CSREP $\Rightarrow$ CDHP as follows:

Given an CSREP-oracle A_3, on input g, g^a for $a \in \mathbb{Z}_p^*$, A_3 output $g^{a^{\frac{1}{2}}}$, if a is a quadratic residue modulo p. Otherwise, output $\perp$. According to theorem 1, computing CDHP in G is polynomial time equivalent to computing Squ-CDHP and Inv-CDHP in G. The following we want to compute g^{a^2} from g and g^a for $a \in \mathbb{Z}_p^*$ using A_3.

If $p = 4k - 1$, for some $k \in \mathbb{Z}$, then $p + 1$ is divisible by 4. If a is a quadratic residue modulo p, we have

$$a^{\frac{1}{2}} \equiv \pm\, a^{\frac{p+1}{4}} \pmod{p}$$

This is because $a^{\frac{p-1}{2}} \equiv 1 \pmod{p}$, then

$$\left(a^{\frac{1}{2}}\right)^2 \equiv a^{\frac{p+1}{2}} \equiv a^{\frac{p-1}{2}} a \equiv a \pmod{p}$$

We have $g^{a^{\frac{1}{2}}} = g^{\pm a^{\frac{p+1}{4}}} = A_3(g, g^a)$.

1). Let $\frac{p+1}{4} = k$. Since $g^{a^{\frac{1}{2}}} = g^{\pm a^k} = A_3(g,\ g^a)$, if $k^i \equiv 2^j \pmod{p-1}$, and i, j are polynomial in $log\ p$ for some $i, j \in \mathbb{Z}$, then we can compute $g^{a^{k^i}}$ from $(g,\ g^a)$, and g^{a^2} from $(g,\ g^{a^{2^j}} = g^{a^{k^i}})$ by iteratively calling oracle A_3 i and $(j-1)$ times, respectively.

When a is not a quadratic residue modulo p, then $A_3(g, g^a)$ output $\perp$. In this case, we have

$$(-a)^{\frac{1}{2}} \equiv \pm\, (-a)^{\frac{p+1}{4}} \pmod{p}$$

then,

$$g^{(-a)^{\frac{1}{2}}} \equiv g^{\pm(-a)^{\frac{p+1}{4}}} \equiv g^{\pm(-a)^k} \equiv g^{\pm(-1)^k a^k} = A_3(g, g^{-a})$$

2). For $k \equiv 2^j + 1 \pmod{p-1}$, then we can compute

$$g^{a^{2^j+2^{j-1}+\ldots+2+1+1}} = g^{a^{(2^{j+1}-1)+1}} = g^{a^{2^{j+1}}}$$

from (g, g^a) by iteratively calling oracle A_3 $j+1$ times, and then we can compute g^{a^2} from $g^{a^{2^{j+1}}}$ by iteratively calling oracle A_3 j times. So, we can compute g^{a^2} by iteratively calling oracle A_3 $2j+1$ times.

3). For $k \equiv (2^j - 2^{j-1} + 1) \pmod{p-1}$, we have

$$a^{\frac{1}{2}} \equiv a^k \equiv a^{2^j - 2^{j-1}+1} \equiv a^{2^j - 2^{j-1}} a \pmod{p}$$

then

$$(a^{\frac{1}{2}})^{\frac{1}{2}} \equiv a^{2^{j-1}-2^{j-2}} a^{2^j-2^{j-1}+1} \equiv a^{2^j-2^{j-2}} a \pmod{p}$$

so we can compute $g^{a^{2^j}}$ from (g, g^a) by iteratively calling oracle A_3 j times, and then we can compute g^{a^2} from $g^{a^{2^j}}$ by iteratively calling oracle A_3 $j-1$ times. So, we can compute g^{a^2} by iteratively calling oracle A_3 $2j-1$ times.

4). Similar to above method, for $k \equiv 2^{2j} \pm 2^j + 1 \pmod{p-1}$, we can compute g^{a^2} by iteratively calling oracle A_3 $8j+3$ times.

5). For $k \equiv 2^{2j+1} \pm 2^j + 1 \pmod{p-1}$, we can compute g^{a^2} by iteratively calling oracle A_3 $3j+2$ times.

For the case of $k^i \equiv -2^j$, $k \equiv -(2^j+1)$, $k \equiv -(2^j - 2^{j-1} + 1)$, $k \equiv -(2^{2j} \pm 2^j + 1)$ and $k \equiv -(2^{2j+1} \pm 2^j + 1) \pmod{p-1}$, using above method, we can get $g^{a^{-2}}$ by iteratively calling oracle A_3, so we can get $g^{a^{-1}}$ by iteratively calling oracle A_3 one more time, this is the Inv-CDHP.

Thus, for $p = 4k-1$, $k^i \equiv \pm 2^j \pmod{p-1}$, or $k \equiv \pm(2^j+1) \pmod{p-1}$, or $k \equiv \pm(2^j - 2^{j-1}+1)$, or $k \equiv \pm(2^{2j} \pm 2^j + 1)$, or $k \equiv \pm(2^{2j+1} \pm 2^j + 1) \pmod{p-1}$ and i, j are polynomial in $\log p$ for some $k, i, j \in \mathbb{Z}$, using CSREP-oracle we can solve Squ-CDHP or Inv-CDHP in G. Which is polynomial time equivalent to computing the CDHP in G. □

Note 1. For simplicity in Theorem 2 and the rest of the paper, we use the expression $x \equiv \pm y \pmod{z}$ to denote $x \equiv y \pmod{z}$ or $x \equiv -y \pmod{z}$. Similarly, we make the expression $x \equiv g^{\pm y} \pmod{z}$ to denote $x \equiv g^y \pmod{z}$ or $x \equiv g^{-y} \pmod{z}$.

Note 2. Because $a^{\frac{1}{2}}$ can always be expressed as a polynomial $f(a)$ of a, then $g^{a^{\frac{1}{2}}} = g^{f(a)}$ which can always be computed by CDHP-oracle and Squ-CDHP-oracle. That is, for any prime order p, we have CDHP ⇒ CSREP.

To check the correctness of the Theorem 2, we provide the following simple example.

Example 1. For instance, for a cyclic group G with prime order $p = 9719$, then $k = \frac{p+1}{4} = 2430$. Therefore, we have $k^i \equiv 2^j \pmod{p-1}$, with $i = 1$ and $j = 27$.

If a is a quadratic residue modulo p, then $g^{a^{\frac{1}{2}}} = g^{\pm a^k} = A_3(g,\ g^a)$, we can compute g^{a^2} by $A_3(g,\ g^{a^{2^{27}}} = g^{a^k})$ by iteratively calling oracle A_3 26 times.

On the other hand, if a is not a quadratic residue modulo p, we have

$$g^{(-a)^{\frac{1}{2}}} = g^{\pm(-a)^{\frac{p+1}{4}}} = g^{\pm(-a)^k} = g^{\pm(-1)^k a^k} = A_3(g, g^{-a}).$$

Therefore, we can always compute g^{a^2} or $g^{(-a^2)}$ by iteratively calling oracle A_3, which means Squ-CDHP in G can be solved in polynomial time.

Till now, we have proved that for about half of all the prime numbers, that is, those $p \equiv 3 \pmod 4$, the CSREP is polynomial time equivalent to the CDHP in G. For the other primes, such as $p \equiv 1 \pmod 4$, it needs to be divide into several cases and can be analyzed similarly.

For example, for the case $p \equiv 5 \pmod 8$, then $p + 3$ is divisible by 8. If $a^{\frac{p-1}{4}} \equiv 1 \pmod p$, then we have

$$a^{\frac{1}{2}} \equiv a^{\frac{p+3}{8}} \pmod p.$$

That is $g^{a^{\frac{1}{2}}} = g^{a^{\frac{p+3}{8}}} = A_2(g, g^a)$. Let $\frac{p+3}{8} = k$ for some $k \in \mathbb{Z}$, then $p = 8k - 3$. Since $g^{a^{\frac{1}{2}}} = g^{a^k} = A_2(g, g^a)$, if $k^i \equiv \pm 2^j \pmod{p-1}$, and i, j are in $O(\log p)$ for some $i, j \in \mathbb{Z}$, then we can compute $g^{a^{k^i}}$ from g, g^a and g^{a^2} from g, $g^{a^{2^j}} = g^{a^{k^i}}$ by iteratively calling oracle A_2 i and $(j-1)$ times, respectively.

In fact for the rest primes, such as $p \equiv 5 \pmod 8$, while $a^{\frac{p-1}{4}} \neq 1 \pmod p$, and $p \equiv 1 \pmod 8$, it can be analyzed similarly but with more complicated expressions. The key point is that $a^{\frac{1}{2}}$ can always be expressed as a polynomial $f(a)$ of a, then $g^{a^{\frac{1}{2}}} = g^{f(a)}$ which can be computed by the CSREP-oracle in polynomial time.

It follows immediately from Theorem 2, if $p = 2^n - 1$, for some $n \in \mathbb{Z}$, then $a^{\frac{p+1}{4}} = a^{2^{n-2}}$. We can compute g^{a^2} from g, $g^{\pm a^{2^{n-2}}}$ by iteratively call CSREP-oracle $(n-3)$ times. That is we have the following corollary.

Corollary 1. *Let G be a cyclic group of prime order p and let $g \in G$ be generator of G. If $p = 2^n - 1$, for some $n \in \mathbb{Z}$, then computing CSREP in G is polynomial time equivalent to computing CDHP in G.*

Therefore, for those cyclic groups whose order is a Mersenne prime, computing CSREP is polynomial time equivalent to computing CDHP in the group.

Moreover, for Theorem 2, we can simplify the expression $k^i \equiv \pm 2^j \pmod{p-1}$ for certain integer i, j and k even further. More precisely, we have the following corollary.

Corollary 2. *Let G be a cyclic group of prime order p and let $g \in G$ be generator of G. If $2^i \equiv \pm 2^{2i+j} \pmod{p-1}$, and i, j are polynomial in $\log p$ for some $i, j \in \mathbb{Z}$,*

then computing CSREP in G is polynomial time equivalent to computing CDHP in G.

Proof. According to Theorem 2, if $p = 4k - 1$, $k^i \equiv \pm 2^j \pmod{p-1}$, and i, j are polynomial in $\log p$ for some $k, i, j \in \mathbb{Z}$, then computing CSREP in G is polynomial time equivalent to computing CDHP in G.

Because $k = \frac{p+1}{4}$, then $(\frac{p+1}{4})^i \equiv \pm 2^j \pmod{p-1}$. Hence

$$(p+1)^i \equiv \pm 2^{2i+j} (\text{mod } p-1).$$

that is,

$$2^i \equiv \pm 2^{2i+j} (\text{mod } p-1).$$

Thus, if $2^i \equiv \pm 2^{2i+j} \pmod{p-1}$, and i, j are polynomial in $\log p$ for some $i, j \in \mathbb{Z}$, then computing CSREP in G is polynomial time equivalent to computing CDHP in G. □

In fact for the above Example 1, because $2^i \equiv 2^{2i+j} \pmod{p-1}$ with $i = 1$ and $j = 27$, we can compute g^{a^2} or $g^{(-a^2)}$ by iteratively calling oracle A_3, which confirms Corollary 2.

Generally, to prove the equivalence relationship between CSREP and CDHP for certain groups, Corollary 2 is always more efficient than Theorem 2. That is, it is more efficient to find the corresponding integers i and j, such that $2^i \equiv \pm 2^{2i+j} \pmod{p-1}$.

4 Algorithm

To confirm that there does exist certain primes that satisfy the conditions of Theorem 2 or Corollary 2, we need more efficient algorithm. That is, to find prime p, such that $p = 4k - 1$, $k^i \equiv \pm 2^j \pmod{p-1}$, and i, j are polynomial in $\log p$ for some $k, i, j \in \mathbb{Z}$. For a prime p, if there exist i, j such that $k^i \equiv \pm 2^j \pmod{p-1}$, then such i, j are usually very large, they are about $O(p)$, not $O(\log p)$. So, given p, it is not easy to find such i, j.

Notice that $k^i \equiv \pm 2^j \pmod{p-1}$, then k must be even, say $k = 2m$ for certain integer m. Because $p = 4k - 1$, then $p = 8m - 1$.

Furthermore, the equation $k^i \equiv 2^j \pmod{p-1}$ with $i = 1$ is equivalent to $2m \equiv 2^j \pmod{8m-2}$, that is, $2^j = n(8m-2) + 2m$ for some integer n. Therefore, we have

$$m = \frac{2^j + 2n}{8n+2} \quad \text{for certain integers } m, n, j.$$

Then the question of finding primes p that satisfy the conditions of Theorem 2 or Corollary 2, is equivalent to find the above integers m, n, j such that $p = 8m - 1$ is a prime. More precisely, we have the following Algorithm 1.

Algorithm 1. Finding primes for Theorem 2 with λ bits

Require: $J1, J2$: the range of j; N: the search range of n.
Ensure: The prime p.

```
for j = J1 to J2 do
  for n = 1 to N do
    a ← 2^j + 2n
    b ← 8n + 2
    if b divides a then
      m ← a/b
      p ← 8m − 1
      if (p is prime) and p ∈ (2^(λ−1), 2^λ) then
        return p
      end if
    end if
  end for
end for
```

In fact, similar discussions and algorithms also work for the cases of $k \equiv \pm(2^j + 1)$, $k \equiv \pm(2^j - 2^{j-1} + 1)$, $k \equiv \pm(2^{2j} \pm 2^j + 1)$ and $k \equiv \pm(2^{2j+1} \pm 2^j + 1)$ mod $p - 1$.

The efficiency of Algorithm 1 determined by the probabilities of $\Pr(8n + 2$ divides $2^j + 2n)$ and $\Pr(8m - 1$ is a prime). Our experiments show that by properly setting search ranges for j and n, one can always get certain primes by Algorithm 1. For instance, we try to find a prime of 160 bits satisfies Theorem 2 using Algorithm 1 as follows.

Example 2. We set $J1 = 170$, $J2 = 175$, $N = 2^{15}$, then we get the prime as

$$p = 1161234830679639844419348332931964840524856273327.$$

then

$$k = 290308707669909961104837083232991210131214068332.$$

Moreover, we have

$$k \equiv 2^{173} \pmod{p - 1}.$$

Then, according to Theorem 2 in the group G of order p, computing CSREP is polynomial time equivalent to computing CDHP. Furthermore, Maurer and Wolf [8,9,10,11] proved that, for every cyclic group G with prime order p, if there exists an elliptic curve over $\mathbb{F}_p$ with smooth order then computing DLP and CDHP in G are equivalent.

Accordingly, for this example we find the auxiliary elliptic curve $\mathbb{E}$ defined by $y^2 = x^3 + ax + b$ over $\mathbb{F}_p$ with smooth order as follows,

$$a = -3.$$

$$b = 1113817040494081862447257898872757433359527021918.$$

$$r = 5 \cdot 181 \cdot 521 \cdot 68437 \cdot 509477 \cdot 68844703 \cdot 82113671 \cdot 102914341 \cdot 121410517.$$

where r is the order of $\mathbb{E}$, for which the smoothness bound less than 2^{27}.

Therefore for group G with prime order p such as stated in Corollary 1 and 2, the computation of the CSREP, CDHP and DLP are all polynomial time equivalent.

5 Conclusion

In this paper, we studied the relations among variants of computational Diffie-Hellman problem, and examined the complexity of a new CDHP variant: CSREP. We proved that under certain conditions the CSREP is also polynomial time equivalent to the CDHP. We also provided the witness that under proper conditions, the DLP, CDHP and CSREP may be polynomial time equivalent with respect to the computational reduction for the first time in the literature. Therefore, one can take the advantages of all these CDHP variants to design cryptography schemes and protocols.

As a further work, we would like to extend our result to a more general situation. In particular, we would like to further explore the possibility of applying the CSREP, in the groups where it is equivalent to the CDHP, to the development of new cryptography schemes.

References

1. Bao, F., Deng, R.H., Zhu, H.: Variations of Diffie-Hellman Problem. In: Qing, S., Gollmann, D., Zhou, J. (eds.) ICICS 2003. LNCS, vol. 2836, pp. 301–312. Springer, Heidelberg (2003)
2. Burmester, M., Desmedt, Y.G., Seberry, J.: Equitable key escrow with limited time span (or, how to enforce time expiration cryptographically). In: Ohta, K., Pei, D. (eds.) ASIACRYPT 1998. LNCS, vol. 1514, pp. 380–391. Springer, Heidelberg (1998)
3. Diffie, W., Hellman, M.: New Directions in cryptography. IEEE Transactions on Information Theory 22, 644–654 (1976)
4. ElGamal, T.: A public-key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory 31, 469–472 (1985)
5. FIPS 186-2, Digital signature standard, Federal Information Processing Standards Publication 186-2 (February 2000)
6. Konoma, C., Mambo, M., Shizuya, H.: Complexity analysis of the cryptographic primitive problems through square-root exponent. IEICE Trans. Fundamentals E87-A(5), 1083–1091 (2004)
7. Konoma, C., Mambo, M., Shizuya, H.: The computational difficulty of solving cryptographic primitive problems related to the discrete logarithm problem. IEICE Trans. Fundamentals E88-A(1), 81–88 (2005)
8. Maurer, U.M.: Towards the equivalence of breaking the diffie-hellman protocol and computing discrete logarithms. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 271–281. Springer, Heidelberg (1994)
9. Maurer, U.M., Wolf, S.: Diffie-hellman oracles. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 268–282. Springer, Heidelberg (1996)
10. Maurer, U.M., Wolf, S.: The Relationship Between Breaking the Diffie-Hellman Protocol and Computing Discrete Logarithms. SIAM J. Comput. 28(5), 1689–1721 (1999)

11. Maurer, U.M., Wolf, S.: The Diffie-Hellman protocol. Designs Codes and Cryptography 19, 147–171 (2000)
12. McCurley, K.: The discrete logarithm problem. In: Cryptology and Computational Number Theory. Proceedings of Symposia in Applied Mathematics, vol. 42, pp. 49–74 (1990)
13. Muzereau, A., Smart, N.P., Vrecauteren, F.: The equivalence between the DHP and DLP for elliptic curves used in practical applications. LMS J. Comput. Math. 7(2004), 50–72 (2004)
14. Peralta, R.: A simple and fast probabilistic algorithm for computing square roots modulo a prime number. IEEE Trans. on Information Theory 32(6), 846–847 (1986)
15. Pfitzmann, B., Sadeghi, A.-R.: Anonymous fingerprinting with direct non-repudiation. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 401–414. Springer, Heidelberg (2000)
16. Pohlig, S.C., Hellman, M.E.: An improved algorithm for computing logarithms over GF(p) and its cryptographic significance. IEEE-Transactions on Information Theory 24, 106–110 (1978)
17. Sadeghi, A.-R., Steiner, M.: Assumptions related to discrete logarithms: Why subtleties make a real difference. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 244–261. Springer, Heidelberg (2001)
18. Schnorr, C.P.: Efficient signature generation by smart cards. Journal of Cryptology 4, 161–174 (1991)
19. Zhang, F., Chen, X., Susilo, W., Mu, Y.: A new signature scheme without random oracles from bilinear pairings. In: Nguyên, P.Q. (ed.) VIETCRYPT 2006. LNCS, vol. 4341, pp. 67–80. Springer, Heidelberg (2006)
20. Zhang, F., Chen, X., Wei, B.: Efficient designated confirmer signature from bilinear pairings. In: ASIACCS 2008, pp. 363–368 (2008)

Author Index

Abe, Masayuki 1

Beimel, Amos 11
Belfiore, Jean-Claude 47
Bernstein, Daniel J. 62, 81

Cohen, Gérard 263

Fan, Xiubin 109
Farràs, Oriol 99
Feng, Dengguo 109
Feng, Xiutao 109

Garay, Juan 126
Givens, Clint 126
Grassl, Markus 142

Hanrot, Guillaume 159
Hayasaka, Kenichiro 191
Høholdt, Tom 201
Hu, Lei 246
Huang, Ming-Deh 213

Justesen, Jørn 201

Kiayias, Aggelos 223
Kohel, David 238

Lange, Tanja 81
Li, Jie 246

Mesnager, Sihem 263

Oggier, Frédérique 47
Ohkubo, Miyako 1
Ostrovsky, Rafail 126

Padró, Carles 99
Peters, Christiane 81
Pujol, Xavier 159

Schwabe, Peter 81
Solé, Patrick 47
Stehlé, Damien 159

Takagi, Tsuyoshi 191

Wang, Ping 283
Wu, Chuankun 109

Yekhanin, Sergey 273

Zeng, Xiangyong 246
Zhang, Fangguo 283
Zhang, Wentao 109